Chemiosmotic Chips: Protonic Computation Beyond Electronics, From Gradient Batteries to Entrainment Architectures

Douglas C. Youvan

doug@youvan.com

www.youvan.ai

December 19, 2025

Semiconductor electronics achieved its dominance by turning charge flow into reliable logic, memory, and amplification on dry, crystalline substrates. Yet biology solved an adjacent problem long ago: it learned to store, route, and spend energy using proton gradients across membranes. Chemiosmosis is not merely a biochemical curiosity; it is a disciplined physical method for building potential energy, controlling dissipation, and coordinating distributed work. This paper asks a deliberately provocative question: what would it mean to build a chip whose native state variables are not voltages and electron densities, but delta_pH and delta_psi—chemiosmotic potentials—computed, stored, and routed on purpose? We outline a path from today's "protonics as a component" toward "chemiosmosis as an operating system." The core idea is an addressable array of sealed compartments, membranes, and pump/channel primitives tha can be written with light, coupled through gated pathways, and stabilized by restoration protocols. Such systems would not compete with CMOS on GHz digital logic; they would compete where CMOS struggles: analog dynamics, statefulness, co-located storage and computation, chemical sensing, and gradient-native learning. We propose a taxonomy of architectures, identify the minimal primitives, and define engineering signatures that distinguish a true chemiosmotic chip from a proton-influenced electronic device.

Keywords: chemiosmosis, protonics, proton gradient, membrane potential, delta_pH, delta_psi, proton-motive force, bacteriorhodopsin, proteorhodopsin, nanofluidics, iontronics, chemimemristor, neuromorphic hardware, gradient battery, reconfigurable canal networks, hybrid bioelectronic systems, restoration protocols, drift control, energy transduction, entrainment.

A collaboration with GPT-5.2-Thinking. 60 pages. CC4.0.

Outline

1. Framing the Question: From CMOS to Chemiosmosis
1.1 What “chemiosmotic chip” means and what it does not mean
1.2 Why biology chose gradients: energy storage, routing, and local work
1.3 The engineering wager: chemistry as state, not merely as noise

2. Minimal Primitives for Chemiosmotic Computation
2.1 The closed compartment as the irreducible “cell” of the chip
2.2 The membrane as boundary condition: selectivity, leakage, and capacitance
2.3 Light-driven pumps as writers: orientation, photon budget, and throughput
2.4 Gated channels as switches: valves, thresholds, and rectification
2.5 Sensors and transducers: reading delta_pH and delta_psi without collapsing state

3. State Variables and the Chemiosmotic “Instruction Set”
3.1 Proton-motive force as the supply rail: combining delta_pH and delta_psi
3.2 Gradient write operations: charging, partial charging, and patterned illumination
3.3 Gradient read operations: non-destructive sensing and differential readout
3.4 Coupling operations: diffusion, gated transfer, and impedance matching
3.5 Reset and restore: why chemiosmotic systems require explicit housekeeping

4. Architecture I: On-Chip Chemiosmotic Battery Arrays
4.1 Addressable micro-batteries: nanocavities, vesicle arrays, and sealing strategies
4.2 Power domains as gradient domains: local rails, isolation, and load attachment
4.3 Loads beyond ATP: proton-coupled work elements and chemical actuators
4.4 Performance envelope: energy density, leakage limits, and duty cycles

5. Architecture II: Complementary Protonic Logic With H+ and OH- Networks
5.1 Why complementary logic matters: restoration, noise margin, and static dissipation
5.2 Carrier symmetry and neutrality: how to avoid “acid accumulation” failure
5.3 Gate primitives: push–pull chemiosmotic switching and differential thresholds
5.4 Candidate material stacks: hydrated polymers, oxides, and selective membranes

6. Architecture III: Protonic FPGA via Reconfigurable Canal Networks
6.1 Channels as wires: dispersion, crosstalk, and spatial computing constraints
6.2 Switch fabric: voltage-gated, light-gated, and chemically programmed routing
6.3 Buffers as capacitors: weak acids/bases, reservoirs, and time constants
6.4 Compilation concept: mapping tasks onto canal graphs and scheduling restoration

7. Architecture IV: Gradient-Native Neuromorphic Chips
7.1 Neurons as thresholds in chemical fields: integrate-and-fire in delta_pH space
7.2 Synapses as buffering capacity: weight storage as local chemistry
7.3 Metaplasticity and volatility: embracing drift as a computational feature
7.4 Learning rules: timing, propagation delays, and local homeostasis

8. Architecture V: Hybrid Chemiosmotic System-on-Chip
8.1 Why hybrid wins early: electronics for timing, calibration, and I/O
8.2 The transducer layer: pH-sensitive FETs, redox bridges, capacitive couplers
8.3 Refresh as a design principle: chemical DRAM and periodic re-zeroing
8.4 Security and integrity: preventing contamination, spoofing, and state hijacking

9. Manufacturing and Integration Pathways
9.1 CMOS compatibility vs bio-MEMS stacks: what “compatible” should mean
9.2 Membrane placement and orientation at scale: templating, self-assembly, lithography
9.3 Sealing and leak testing: metrology for pmf retention and channel defects
9.4 Reliability physics: the protonic analogs of electromigration and dielectric breakdown

10. Performance Metrics and Benchmark Tasks
10.1 Core metrics: pmf magnitude, retention time, write energy, endurance, variability
10.2 System metrics: fan-out, routing loss, refresh overhead, and thermal coupling
10.3 Benchmark tasks: sensing-to-inference loops, reservoir computing, adaptive control
10.4 What would count as a decisive demo: three “signature” experiments

11. Failure Modes, Limits, and Repairs
11.1 Leak-dominated scaling: why defects win unless architecture changes
11.2 Drift and hysteresis: distinguishing useful memory from unbounded instability
11.3 Biofouling and contamination: the hidden enemy of long-lived gradients
11.4 Repair protocols: reconditioning membranes, recalibration, redundancy, and quarantine

12. Conceptual Synthesis: Chemiosmosis as an Operating System
12.1 The OS metaphor: scheduling, refresh, isolation, and privilege levels
12.2 Entrainment architectures: patterned drives and state locking in gradient space
12.3 When analogy becomes physical: alignment-as-entrainment in a chemiosmotic substrate

13. Conclusion: Beyond Electronics Without Replacing Electronics
13.1 Restating the thesis: gradients as first-class computational state
13.2 The practical takeaway: build hybrid stacks, prove signature demos, then scale
13.3 Closing reflection: a boundary-driven physics of computation

14. References
14.1 Chemiosmosis and membrane bioenergetics foundations
14.2 Proton transport physics and protonic materials
14.3 Iontronics, electrochemical transistors, and neuromorphic devices
14.4 Nanofluidics, membrane integration, and hybrid bioelectronic platforms
14.5 Benchmarking, reliability, and systems design methodologies

1. Framing the Question: From CMOS to Chemiosmosis

1.1 What “chemiosmotic chip” means and what it does not mean

A chemiosmotic chip is not simply an electronic device influenced by ions, nor is it a biochemical sensor attached to conventional circuitry. It is a computational substrate whose *native, intentional state variables* are chemiosmotic quantities: proton concentration differences, membrane potentials, and their coupled electrochemical fields. In such a system, delta_pH and delta_psi are not side effects to be measured and translated back into voltages; they *are* the stored state, the carriers of energy, and the medium through which operations occur.

Equally important is what the term does not mean. A chemiosmotic chip is not an attempt to recreate a living cell on silicon, nor is it a proposal to replace CMOS for high-speed digital logic. It does not require enzymes, metabolism, replication, or genetic control. It is not “wet computing” in the vague sense of chemistry standing in for electronics. Rather, it is a disciplined engineering effort to harness one specific physical mechanism—gradient formation across boundaries—and to treat it with the same seriousness that electronics treats voltage, charge, and capacitance.

The distinction matters because much prior work in protonics and iontronics uses chemistry instrumentally: ions modulate an electronic channel, drift is tolerated as an annoyance, and stability is borrowed from the surrounding CMOS. A chemiosmotic chip, by contrast, accepts that gradients are slow, leaky, history-dependent, and spatially distributed—and then asks how to compute *because of*, not in spite of, those properties.

1.2 Why biology chose gradients: energy storage, routing, and local work

Biology’s reliance on chemiosmosis was not an arbitrary evolutionary choice; it is a consequence of physics at small scales. In aqueous, thermally noisy environments, storing energy as bulk chemical bonds is difficult without complex machinery, and storing it as electric charge alone is limited by leakage and screening. Gradients across membranes offer a third option: energy stored as a boundary condition.

A chemiosmotic gradient simultaneously satisfies three requirements that are hard to meet otherwise. First, it enables energy storage that is spatially localized yet globally accessible within a compartment. Second, it allows routing: energy can be delivered to specific sites by opening channels or coupling reactions locally, rather than distributing power uniformly. Third, it supports local work. The gradient does not merely exist; it can directly bias reactions, drive transport, or actuate molecular machines exactly where needed.

This triad—storage, routing, and local work—is precisely what electronic power distribution networks struggle to provide efficiently at fine granularity. Electronics excels at fast signaling but

pays a high price to move energy across a chip and to isolate it into small domains. Biology, constrained by slow carriers and noisy environments, inverted the problem: it built systems where energy is slow to create but cheap to hold and selectively expensive to release. The result is extraordinary efficiency at the cost of speed.

A chemiosmotic chip takes this biological insight seriously. It treats gradients not as a curiosity but as a proven physical solution to distributing energy and state across dense, heterogeneous systems where locality matters more than clock rate.

1.3 The engineering wager: chemistry as state, not merely as noise

The central wager of chemiosmotic computation is that chemistry can be elevated from a source of variability to a source of structured state. Traditional engineering has learned to fear chemistry because it introduces drift, hysteresis, aging, and irreversibility. CMOS works so well precisely because it suppresses chemical degrees of freedom, freezing matter into stable lattices where electrons can be counted and controlled.

A chemiosmotic chip reverses this logic. It assumes that statefulness, hysteresis, and relaxation are not bugs but features—provided they are bounded, observable, and restorable. Protonation states, buffer capacities, hydration levels, and membrane potentials are slow variables with memory. They naturally integrate inputs over time, decay on characteristic scales, and respond nonlinearly to thresholds. These are precisely the properties desired in analog computation, adaptive systems, and neuromorphic architectures.

The wager is not naive. Chemistry is allowed to be stateful only within explicit architectural constraints. Compartments must be sealed. Leakage must be measured and budgeted. Restoration must be designed, not hoped for. Drift must be countered by periodic refresh or homeostatic feedback. In other words, chemistry must be given the same architectural discipline that electronics gives to charge.

If this wager pays off, the reward is a new class of computing substrates that operate comfortably in regimes where CMOS is strained: low-power continuous operation, intimate coupling to physical and chemical environments, and computation that unfolds in time rather than being clocked into discrete instants. Chemiosmosis then becomes not a metaphor but a concrete engineering alternative—a boundary-driven physics of computation that complements, rather than competes with, electronics.

2. Minimal Primitives for Chemiosmotic Computation

2.1 The closed compartment as the irreducible “cell” of the chip

The first primitive is not a molecule, a pump, or a channel. It is *closure*. Chemiosmotic computation begins the moment “inside” and “outside” become physically meaningful, because only then can a gradient accumulate rather than instantly dissipate. The closed compartment is therefore the irreducible “cell” of the chip: the smallest unit in which delta_pH and delta_psi can become persistent state variables rather than transient fluctuations.

From an engineering perspective, the compartment is a state container with a volume, a surface area, and a leakage boundary. Those three numbers set the achievable dynamic range and time constants. Volume determines how many protons (or buffering equivalents) are required to move delta_pH by a measurable amount. Surface area determines how many pumps or channels can act on that state in parallel, and it also determines capacitive coupling for delta_psi. Leakage—through membrane defects, imperfect seals, or unintended shunt pathways—determines retention time. In a chemiosmotic chip, retention time is not a vague “stability” goal; it is a design parameter analogous to DRAM refresh intervals, except now the stored state is chemical and electrostatic rather than purely electrostatic.

The compartment also defines addressability. A single sealed cavity is a curiosity; an array of sealed cavities is an architecture. When compartments are arrayed, each becomes an addressable node whose state can be written by pumps, coupled through channels, and read by sensors. The geometry determines whether the system behaves like isolated registers, a coupled field, or a hybrid of both. In this sense, the “cell” is not a metaphor borrowed from biology; it is the substrate’s minimal unit of locality and isolation, without which computation has no place to reside.

2.2 The membrane as boundary condition: selectivity, leakage, and capacitance

If the compartment is the container of state, the membrane is the *law* that governs state. It is the boundary condition that determines what crosses, what is blocked, and how electrical potential is stored. A membrane in chemiosmotic computation must do three things at once: provide selectivity, limit leakage, and behave as a capacitor.

Selectivity is the membrane’s logic layer. It determines whether the gradient is principally a proton gradient (delta_pH), an electrical gradient (delta_psi), or a controlled combination. Selectivity is not merely about letting protons through; it is about controlling *which other ions* are allowed to follow, because uncontrolled counterion flux will neutralize the electrical component and collapse useful potential. The membrane must therefore be engineered as a selective barrier with known permeability for relevant carriers. In practical systems, “proton

selective" is usually an approximation; real membranes leak water, ions, and charge in complicated ways. The key is not perfection but predictability and budgeted loss.

Leakage is the membrane's enemy and its teacher. Every chemiosmotic system is ultimately limited by unintended conduction paths: pores, grain boundaries, pinholes, adsorption layers, and interface defects. The chemiosmotic chip must treat leakage as a first-class quantity, measured per node, tracked over time, and managed by design. This leads naturally to architectures that tolerate leakage through refresh, redundancy, and error-correcting layouts. A membrane that leaks slowly is not useless; it simply imposes a time constant. The difference between "device" and "toy" is whether that time constant is stable enough to engineer around.

Capacitance is the hidden gift of the membrane. Any thin dielectric boundary between two conductive media stores charge. In chemiosmosis, this means delta_psi can be stored as charge separation across the membrane, coupled to the ionic distributions on each side. This gives chemiosmotic chips a powerful design lever: they can trade delta_pH and delta_psi, shaping which component carries the burden of storage and which carries the burden of signaling. Capacitance also sets switching energetics. In electronic CMOS, C*V^2 dominates dynamic energy. In chemiosmotic computation, an analogous term exists, but it is entangled with ionic motion, screening, and chemical buffering. The membrane is therefore simultaneously a capacitor, a resistor, and a selective filter—a boundary that makes the entire regime possible.

2.3 Light-driven pumps as writers: orientation, photon budget, and throughput

In a chemiosmotic chip, pumps are "writers": they inject free energy into a node's state variables, turning illumination into controlled changes in delta_pH and delta_psi. Light-driven proton pumps are particularly attractive because they collapse the write pathway into a single physical event: photon absorption triggers conformational change, which biases proton movement across the boundary. No external redox reagents are required. This creates a clean separation between energy input (light) and state evolution (gradient accumulation).

Orientation is the first decisive issue. A pump that faces the wrong way writes the opposite sign. A mixture of orientations writes noise. Biology solves this by complex membrane assembly. Engineering must solve it by templating, patterned anchoring, electric-field alignment, or self-assembly strategies that strongly bias orientation. The orientation problem is not an implementation detail; it is the equivalent of transistor polarity in CMOS. If orientation cannot be controlled, scaling fails because adding more pumps does not reliably increase write strength.

Photon budget is the second issue. A pump is a quantized actuator: one photon produces at most one successful transport cycle per excitation event, and real systems waste photons through absorption that does not yield transport. Therefore, the write energy per state update

is determined by both optical coupling and pump quantum efficiency. This is not a weakness; it is a design handle. Optical writing enables spatial addressing without wires, multiplexing by wavelength, and parallelism over large arrays. But it also forces you to think like an optical engineer: you must route photons, prevent cross-talk, and manage thermal load. The relevant "clock" is not a GHz electrical pulse but a photon flux and a pump turnover rate.

Throughput is the third issue: the rate at which a node can be charged or modified. Throughput depends on pump density (how many pumps per unit area), turnover rate, and the compartment's buffering capacity. If the compartment is strongly buffered, many protons must be pumped to move delta_pH, which slows writing but increases stability. If the compartment is weakly buffered, small proton numbers create large delta_pH swings, increasing sensitivity but risking noise and saturation. Thus, pump writing is inseparable from compartment chemistry. A chemiosmotic chip is designed by matching pump throughput to buffering and leakage, so that state changes are resolvable, stable, and fast enough for the target task.

2.4 Gated channels as switches: valves, thresholds, and rectification

Channels are the "switches" of chemiosmotic computation. Pumps write gradients; channels route, couple, and discharge them. But a channel is not a transistor. It is closer to a valve whose conductance can be modulated, whose behavior may be nonlinear, and whose state may itself be history-dependent.

Valving is the basic operation: open the pathway and allow proton flow (or counterion flow, depending on design); close it and isolate nodes. This defines connectivity. In an array of compartments, channels determine whether nodes are independent registers, coupled oscillators, or a diffusive field. Even simple valve control enables useful operations: charge sharing (analogous to capacitive coupling), gated discharge (thresholded reset), and routing (moving gradient from one node to another).

Thresholds are how chemiosmotic systems compute. A channel whose opening depends on local delta_pH or delta_psi implements a built-in nonlinearity. This gives you integrate-and-fire behavior without imposing it digitally. You can store a slowly accumulating gradient until it crosses a threshold and triggers a rapid flow event, creating pulse-like dynamics. In neuromorphic terms, the channel becomes the spiking element. In logic terms, it becomes the comparator.

Rectification is the chemiosmotic analog of diode behavior. Biological membranes exploit rectification constantly: certain channels conduct preferentially in one direction due to energy barriers and gating kinetics. In a chemiosmotic chip, rectification matters because it allows directional routing and prevents backflow that would erase stored gradients. Without

rectification, coupling networks become symmetric diffusers that equalize everything. With rectification, they become directed graphs capable of computation over flows.

Finally, channels introduce their own failure modes: unintended leak conductance, stochastic gating, and susceptibility to fouling. But they also provide repair pathways: you can isolate a faulty node by closing its channels, quarantine drift, and reroute around defects. In a sense, channels are both the logic and the fault containment fabric.

2.5 Sensors and transducers: reading delta_pH and delta_psi without collapsing state

In electronics, reading a voltage can often be done with negligible disturbance. In chemiosmotic systems, reading is inherently invasive if it requires moving ions or puncturing boundaries. Therefore, the sensor/transducer layer is the make-or-break primitive: it determines whether chemiosmotic state can be observed and used without being destroyed.

Reading delta_pH can be done chemically or electronically. Chemical readout (fluorescent pH indicators, absorbance dyes, etc.) is powerful for lab demonstrations but often impractical for integrated systems unless optical infrastructure is built in. Electronic readout aims to couple pH to an electrical signal through pH-sensitive interfaces: surface charge changes, field-effect modulation, or capacitive coupling. The guiding principle is that the readout should depend on equilibrium properties (surface potential, protonation state at an interface) rather than requiring bulk transport that discharges the compartment.

Reading delta_psi is similarly subtle. The membrane potential is an electric field across a thin boundary. In principle, it can be sensed capacitively, like measuring the voltage across a capacitor, but in practice any electrode contact risks providing a leakage path. Therefore, delta_psi readout often becomes a question of non-contact coupling: capacitive pickoff, floating-gate style sensing, or differential measurement that references adjacent nodes. The engineering goal is analogous to a high-impedance voltmeter, but now the impedance must be high not only electrically but also ionically and chemically.

Transduction is the bridge to hybrid systems. Even if the core computation is chemiosmotic, useful output often needs to be electrical, optical, or chemical. Transducers can convert gradient energy into electronic signals (pH-gated transistors, redox mediators), optical signals (pH-dependent fluorescence), or mechanical signals (actuation of responsive polymers). The important constraint is to avoid collapsing the stored state: the transducer must either be non-destructive or must consume state in a controlled and budgeted way, like a read that is known to be destructive but is paired with refresh.

The deeper point is that in chemiosmotic computation, “read” is not a passive operation. It is a negotiated interaction with a stored gradient that may be both the signal and the power source. The chip must therefore treat sensing as part of the architecture, with explicit read energy

budgets, read disturbance models, and restoration procedures. A chemiosmotic chip is not fully defined until it can say, precisely: what is state, how is it written, how is it routed, and how is it read without erasing what makes it valuable.

3. State Variables and the Chemiosmotic "Instruction Set"

3.1 Proton-motive force as the supply rail: combining delta_pH and delta_psi

A chemiosmotic chip needs a single unifying concept that plays the role that "Vdd" plays in CMOS: a supply rail that can be stored locally, distributed with intent, and spent to perform operations. In chemiosmosis, that role is played by the proton-motive force (pmf), a composite potential that includes both chemical and electrical contributions. The chemical term is the concentration difference of protons across the membrane (delta_pH). The electrical term is the membrane voltage (delta_psi). Together they define the direction and the available free energy for proton flow. In plain engineering language: pmf is the system's "usable gradient energy per unit charge," and it is the currency that powers state transitions.

The crucial design freedom is that pmf can be partitioned. A node can store more of its potential as delta_pH (chemical storage) or more as delta_psi (electrostatic storage). These are not interchangeable in a trivial way, because they differ in dynamics and failure modes. Delta_psi can change rapidly and can couple strongly through capacitance, but it is sensitive to counterion mobility and leakage paths that short the voltage. Delta_pH can be slower and buffered, with high apparent "capacity," but it can be neutralized by diffusion and chemical buffering mismatch. A good chemiosmotic architecture does not pick one; it deliberately chooses a mix for each layer of function.

This is where the analogy to electronics is most productive. In CMOS, "voltage" is not just a number; it is a relationship between capacitances, leakage, and noise margins. In chemiosmosis, pmf is similarly not just a free-energy number; it is a relationship between membrane capacitance, buffering capacity, ion selectivity, and leakage. Therefore, a chemiosmotic chip defines its supply rail by specifying not only the target pmf, but also the permitted partitioning between delta_pH and delta_psi, the acceptable leak rate, and the refresh interval. "Power" is not globally delivered; it is locally grown, maintained, and spent.

3.2 Gradient write operations: charging, partial charging, and patterned illumination

Writing state in a chemiosmotic chip means establishing a controlled pmf (or changing it) in a specific compartment. The simplest write instruction is charge: drive pumps so that protons are moved across the boundary and the node's pmf rises in magnitude. But even this operation has multiple modes that become distinct "instructions."

Full charging is saturating. It drives the node toward a maximum pmf set by pump stall conditions, leakage equilibrium, buffering limits, and counterion constraints. Full charge is useful for initializing power domains and for writing robust “high” states with maximum energy margin. Yet full charge is also expensive: it consumes time and photons, and it may create strong fields that accelerate leakage or induce unwanted reactions. Therefore, full charge is not the default write; it is a deliberate choice.

Partial charging is the workhorse. It sets pmf to an intermediate level—analogous to writing an analog voltage onto a capacitor. This enables multi-level state and graded computation. Partial charge can be defined by duration of illumination, intensity, wavelength selection, duty-cycle modulation, or feedback control from a sensor layer. Partial charge also enables calibration: write a known increment, measure response, correct drift, repeat. In this way, write operations become programmable rather than binary.

Patterned illumination is where chemiosmotic chips acquire an ability that electronics struggles to emulate: parallel spatial addressing without a dense wiring network. If you can deliver light patterns—through micro-LED arrays, waveguides, scanned beams, or projected images—you can write many nodes at once, and you can write them with spatial gradients. This naturally supports field computation. Instead of thinking of the chip as a collection of registers, you can think of it as a substrate that supports shaped energy landscapes: regions of higher pmf, boundaries between domains, and moving patterns that drive entrainment. Patterned illumination becomes both programming and dynamics: it can write initial conditions and also serve as the forcing function that sustains computation over time.

In a mature instruction set, write operations will include not only “charge” but also “bias,” “clamp,” and “dither.” Bias adds a constant offset to counter drift. Clamp holds pmf in a limited range by intermittent pumping and controlled leakage through gated pathways. Dither uses small oscillatory writes to probe node responsiveness and detect fault conditions. These are not luxuries; they are the methods by which a chemical substrate is made predictable.

3.3 Gradient read operations: non-destructive sensing and differential readout

Reading is not simply observing. Reading is an interaction with a stored gradient, and every interaction risks discharging the very quantity you want to measure. The read instruction set must therefore be designed around non-destructive and minimally destructive operations.

Non-destructive reads are those that sense equilibrium properties without drawing significant ionic current. For delta_pH, this typically means sensing local protonation at an interface—surface charge changes, field-effect modulation, or optical indicators—rather than sampling bulk fluid. For delta_psi, it means capacitive pickoff or floating-gate style sensing that does not provide a conduction path across the membrane. The ideal read behaves like an electrometer:

extremely high impedance, low disturbance, and stable calibration. The practical reality is that "high impedance" must include both electronic and ionic leakage channels; a sensor that is electrically high impedance but chemically leaky is still destructive.

Differential readout becomes central because absolute chemiosmotic values drift. Temperature, hydration, slow chemistry, and aging can shift baseline pH or membrane properties. Differential read compares one node to a reference node, or measures a gradient between paired compartments designed to track common-mode changes. In effect, the chip carries its own reference standards. This is analogous to differential amplifiers, common-mode rejection, and sense amplifiers in memory arrays. Without differential design, the substrate's natural drift will be mistaken for computation.

Another key read primitive is thresholded read. Instead of measuring an analog value precisely, the system can ask: has the node crossed a threshold? This can be done by using a gated channel that opens only beyond a set delta_pH or delta_psi and produces a detectable event. Thresholded read trades precision for robustness and can be closer to how the substrate naturally wants to operate. It also offers a clean interface to digital electronics: a gradient crossing produces a pulse, which electronics can count and route.

Finally, reads have cost. A chemiosmotic instruction set should explicitly budget read energy and read disturbance. Some reads may be destructive by design, akin to latching a value by discharging a capacitor. If so, the instruction set must pair that read with refresh or rewrite. Mature chemiosmotic computing therefore treats read as a first-class operation with defined side effects, not as an assumption.

3.4 Coupling operations: diffusion, gated transfer, and impedance matching

Coupling operations define how state moves and interacts across the chip. Without coupling, each compartment is an isolated battery. With coupling, the array becomes a computational medium.

Diffusive coupling is the default physics: if compartments are connected through a pathway, gradients tend to equalize. Diffusion therefore implements analog averaging, smoothing, and low-pass filtering. In many architectures, this is useful rather than harmful. It can provide the substrate with natural regularization, stabilization of noisy writes, and spatial integration. But diffusion is also the enemy of long-range specificity. Left unstructured, it washes out distinctions.

Gated transfer is coupling under control: open a valve briefly to transfer some gradient "charge" from one node to another, then close it. This enables packetized movement of chemical potential, analogous to charge sharing on capacitors. Gated transfer can implement operations such as accumulate (move small contributions into one node), distribute (broadcast a gradient

from one node to many), and compare (allow two nodes to exchange until a thresholded event occurs). Crucially, gated coupling can be directional if rectification is built into channels. Directionality turns a diffusive substrate into a routed network.

Impedance matching is the subtle operation that separates working systems from fragile ones. In chemiosmotic terms, “impedance” is not a single resistance; it is the effective relationship between conductance, buffering capacity, capacitance, and leakage. When coupling two nodes, if one has high buffering and the other low, a transfer event may produce large swings in one and negligible change in the other. If one node stores mostly delta_psi and the other mostly delta_pH, coupling may lead to unpredictable redistribution. Therefore, chemiosmotic coupling needs matching layers: buffers, intermediate compartments, or controlled conversion between delta_pH and delta_psi. In electronics, impedance mismatch causes reflection and power loss; in chemiosmotic networks, mismatch causes dissipation, loss of controllability, and drift amplification.

A mature instruction set will include explicit coupling primitives: connect, disconnect, transfer_quanta, equilibrate_to_target, and isolate_fault. These primitives are less about logic gates and more about managing a field of potential, in which computation is the orchestrated movement and stabilization of gradients across space and time.

3.5 Reset and restore: why chemiosmotic systems require explicit housekeeping

Every chemiosmotic computation is a contest against entropy and leakage. In CMOS, leakage exists but can be made small enough that many computations complete before it matters, and the digital abstraction tolerates it with margins and error correction. In chemiosmotic substrates, leakage and drift are typically not ignorable; they are native. Therefore, reset and restore are not peripheral operations—they are the operating system.

Reset is the ability to bring a node to a known baseline state. In a gradient system, baseline might mean neutral delta_pH and zero delta_psi, or it might mean a standardized “resting pmf” that serves as the ground state for computation. Reset can be achieved by opening channels to a reservoir, by using reverse pumping, by chemical buffering, or by controlled dissipation. The key is determinism: reset must be faster and more reliable than passive relaxation, otherwise the substrate cannot be scheduled.

Restore is the ongoing maintenance of usable states against drift. It is the chemiosmotic analog of refresh in DRAM, but broader: it includes recalibration of sensors, correction for pump efficiency changes, compensation for slow changes in buffering, and isolation of nodes whose leakage exceeds budget. Restore can be periodic (refresh cycles) or continuous (homeostatic feedback loops). Either way, it must be designed as part of computation, not as a background assumption.

Why is housekeeping so central? Because chemiosmotic computation lives in analog space. Analog systems accumulate small errors. Drift changes baselines. Leakage changes time constants. Without explicit maintenance, the substrate's state becomes uninterpretable. But with maintenance, a powerful new regime opens: the chip becomes a dynamical system whose stability is an engineered property. Computation then includes both the "foreground" task and the "background" discipline that keeps the task meaningful.

In a sense, chemiosmotic chips require an honesty that CMOS can hide. CMOS can pretend that wires are perfect and states are digital. Chemiosmosis cannot. It forces you to specify what "known state" means, how long it remains known, how you detect when it has degraded, and what you do to restore it. This is not a disadvantage; it is the essence of the paradigm. A chemiosmotic instruction set is therefore incomplete unless it includes explicit reset/restore primitives and a scheduling logic that treats maintenance as a costed operation. Only then can gradients become a reliable medium for computation rather than an unstable curiosity.

4. Architecture I: On-Chip Chemiosmotic Battery Arrays

4.1 Addressable micro-batteries: nanocavities, vesicle arrays, and sealing strategies

The chemiosmotic battery array is the most literal interpretation of a "chemiosmotic chip": an addressable grid of sealed compartments that can be charged by light-driven pumps and that store usable potential energy as pmf. Each compartment functions as a micro-battery—not in the electrochemical cell sense of metal electrodes and redox couples, but in the Mitchell sense: a stored electrochemical gradient across a boundary that can later do work.

There are three main physical implementations of the compartment, each defining a different integration path.

Nanocavities are solid-state-defined compartments, fabricated by lithography or etching, then sealed with a membrane spanning an aperture or encapsulating a cavity mouth. The advantage of nanocavities is positional determinism: you know exactly where each battery is, how big it is, and how it couples to on-chip sensors and channels. You also inherit a manufacturing language from microfluidics and MEMS. The disadvantage is that the sealing interface—where membrane meets solid—becomes a dominant leak and failure site. The chip lives or dies by that gasket.

Vesicle arrays are the opposite: compartments formed by self-assembly (liposomes or polymer vesicles) that are then immobilized and addressed on a surface. Their advantage is exquisite membrane quality, because vesicles form closed bilayers naturally, and their curvature can be tuned. Their disadvantage is addressability and uniformity: a million vesicles is easy; a million vesicles each with known orientation, known size distribution, and reliable electrical coupling to

a readout layer is hard. Vesicle arrays therefore tend to begin as high-density ensembles and gradually move toward individually addressable nodes as patterning and attachment chemistry mature.

Hybrid cavity–vesicle architectures try to take the best of both. For example, cavities can serve as docking sites that "host" vesicles or membrane patches, creating a mechanically protected enclosure while retaining biologically derived membrane quality. In effect, the cavity provides the coordinate system, while the vesicle provides the seal.

Across all three, sealing strategies are the real engineering frontier. Sealing is not just "closing" the compartment; it is preventing unintended shunts that quietly erase pmf. Sealing must address multiple leakage modes: ionic leakage through defects, proton leakage through unintended hydrated pathways, and electrical leakage through conductive films or electrodes that accidentally bridge the boundary. Strategies fall into three broad categories: geometric sealing (conformal closure that reduces defect density), chemical sealing (surface functionalization to minimize wettable leak pathways), and architectural sealing (accept leakage but bound it through refresh and redundancy). The important shift is conceptual: a chemiosmotic battery array treats sealing as the analog of transistor yield in CMOS. You can tolerate a certain defect rate, but only if you can map, quarantine, and refresh around it.

Finally, addressability implies an "address decoder" in the physical layer. In electronics, addresses are wires. In chemiosmotic arrays, addresses can be optical (selectively illuminate a node), electrical (apply gating fields locally), or topological (open the specific channels that connect to that node). The simplest, and often the first to work, is optical addressability, because it avoids adding conductive feedthroughs that compromise sealing. A plausible maturation path is therefore: optically write, electrically read; then later add electrical gating for routing and fine control.

4.2 Power domains as gradient domains: local rails, isolation, and load attachment

Once compartments exist, the next architectural leap is to treat them as power domains. In CMOS, power distribution is global and expensive: a few rails feed many gates, and noise isolation becomes a struggle. Chemiosmotic battery arrays invert this: the rails are local and numerous. Each compartment is its own reservoir of potential, and computation can be done by spending local pmf rather than importing energy through long interconnect.

This "local rail" principle changes how one thinks about power integrity. Instead of worrying primarily about IR drop in metal lines, you worry about gradient droop and leakage. Instead of worrying about coupling noise between rails through capacitance, you worry about coupling between compartments through unintended diffusion paths or shared reservoirs. The design

problem is similar in spirit but different in substance: maintain usable margins while densely packing many power domains.

Isolation becomes the central structural feature. Because chemiosmotic power is stored in gradients, any unintended connection is a short circuit. Therefore, isolation is not only a device property (membrane integrity) but also a network property: channels must default closed, inter-compartment coupling must be gated, and shared reservoirs must be carefully controlled. One may even need "firebreak" compartments—sacrificial buffers that absorb leakage and prevent a fault from cascading through the array.

Load attachment is where the battery array stops being an array of potentials and becomes a machine. A load is any element that consumes pmf to do work: moving ions, changing conformation, driving a reaction, actuating a polymer, generating an electrical signal. But attaching a load changes the node's effective impedance, introducing droop and potentially destabilizing neighboring nodes if coupling exists. Therefore load attachment is best treated as a controlled operation: connect_load, consume, disconnect_load. Loads should be attached transiently for computation steps rather than permanently, unless the node is designed as a continuously powered domain.

A particularly powerful possibility is to define gradient domains of different "qualities." Some compartments store high delta_psi for fast switching; others store high delta_pH for stable storage; others serve as buffers. The array then becomes a heterogeneous power fabric, akin to a system-on-chip with multiple voltage domains, but now the domains differ not only in magnitude but in the chemical/electrical partition of pmf. This heterogeneity is not an annoyance; it is a way to exploit the substrate's native degrees of freedom.

4.3 Loads beyond ATP: proton-coupled work elements and chemical actuators

ATP synthase is the canonical biological load, but it is not the most natural load for a chip. It is complex, orientation-sensitive, and optimized for a biological economy of molecules rather than a microfabricated economy of devices. The broader idea is simpler: any mechanism that couples proton flow (or pmf) to a controlled state change can serve as a load. The chip does not need to "make ATP" to be chemiosmotic; it needs to convert gradient energy into an action that is useful for computation or transduction.

One class of loads is ion transport work elements. A proton gradient can drive secondary transporters or engineered antiporters that move other ions. On a chip, this is valuable because it allows chemiosmotic power to reconfigure ionic environments, alter local conductivity, or change the state of other nodes by chemical signaling. In effect, protons become the driver that moves the "information ions."

A second class is electrochemical transducers that convert pmf into electronic signals. For example, a proton gradient can modulate surface charge, shifting a field-effect device, or drive a redox mediator that changes an electrode potential. This is a particularly important load class in hybrid systems: the chemiosmotic domain does the energy storage and analog dynamics; the electronic domain reads out and routes signals. In this case, the "work" done by pmf is to create an electrical observable without requiring direct ionic discharge that would erase state too quickly.

A third class is chemical actuators and responsive polymers. Many polymers and hydrogels undergo swelling, contraction, or stiffness changes in response to pH. If a compartment's delta_pH is used to actuate a membrane flap, valve, or mechanical coupling element, then the gradient directly performs mechanical work. This yields chemiosmotic logic elements where state controls topology: pH changes open or close channels, shift coupling strength, or change the permeability of a barrier. Such loads are attractive because they allow computation to be self-routing: the stored gradient can physically reconfigure the pathways along which gradients move.

A fourth class is catalytic and reaction loads. Proton gradients can bias reaction rates, shift equilibria, and provide controlled microenvironments. A chemiosmotic battery array could therefore drive local catalysis that changes the chemical state of a node—effectively writing memory in a different substrate (polymer protonation state, surface functional group states, redox states) using pmf as the power. This is a route to nonvolatile or semi-volatile memory: the gradient is transient, but it imprints a longer-lived chemical modification.

The point is that "loads" are not a single mechanism. They are a design space. Chemiosmotic battery arrays become computationally interesting when loads are chosen that (i) respond sharply or usefully to pmf, (ii) can be coupled and decoupled under control, and (iii) do not demand biological complexity unless that complexity buys something essential.

4.4 Performance envelope: energy density, leakage limits, and duty cycles

The performance of a chemiosmotic battery array cannot be captured by a single number like "clock speed." Its envelope is defined by energy density, retention, write/refresh costs, and operational duty cycle.

Energy density is constrained by geometry and buffering. Smaller compartments can change delta_pH with fewer transported protons, which increases sensitivity and reduces write energy, but also makes the system susceptible to stochastic fluctuations and saturation. Larger compartments are more stable but require more pumping to achieve the same delta_pH. Meanwhile, delta_psi storage depends on membrane capacitance and the effective ionic screening environment; it can be fast but may be limited by counterion mobility that neutralizes

charge separation. Therefore, energy density is not simply “how much pmf per volume,” but “how much usable pmf can be held with acceptable noise and leakage.”

Leakage limits define retention time. Leakage is the composite of membrane defects, interface shunts, and unintended conductive pathways. In battery arrays, leakage sets an upper bound on useful compute time between refresh cycles. A node with high leakage is not necessarily useless; it can be assigned to fast dynamics roles, or it can be periodically recharged. But the architecture must explicitly manage leakage variability, because the array will not be perfectly uniform. This is analogous to bad cells in memory arrays: you either map them out or assign them roles consistent with their properties.

Duty cycle emerges as a central systems metric. Chemiosmotic nodes may spend substantial time in maintenance: charging, refreshing, calibrating sensors, and compensating drift. The fraction of time spent on “useful computation” versus “housekeeping” depends on leakage rates, pump throughput, and the computational style. A battery array designed for sporadic sensing and adaptive control might tolerate a high maintenance duty cycle because the computation is event-driven and low rate. A battery array designed for continuous neuromorphic dynamics might instead require a steady-state regime where maintenance is integrated into the dynamics via homeostatic feedback, minimizing disruptive refresh cycles.

Finally, the array’s performance must be judged by what it enables that CMOS struggles to do efficiently. The relevant benchmarks are not integer multiplications per second; they are tasks that benefit from local energy storage, analog statefulness, and intimate coupling to chemical or physical environments. Examples include continuous low-power inference from chemical sensors, adaptive control systems that learn slowly in situ, and reservoir-like dynamics where the array’s intrinsic relaxation and coupling provide computation “for free.”

In this architecture, the decisive demonstration is not merely that a node can be charged by light. It is that an array can be charged, held, selectively coupled, used to drive loads, and restored repeatedly—while maintaining interpretable state over time. When that is achieved, the chemiosmotic battery array becomes a true on-chip power-and-state fabric: a computational medium whose currency is gradient energy and whose logic is the controlled creation, movement, and consumption of pmf.

5. Architecture II: Complementary Protonic Logic With H+ and OH- Networks

5.1 Why complementary logic matters: restoration, noise margin, and static dissipation

If chemiosmotic battery arrays are the “power fabric,” complementary protonic logic is the attempt to build a *logic fabric* that inherits the two decisive virtues of CMOS: (i) the ability to

restore signals cleanly and (ii) low static dissipation when nothing is switching. The motivation is straightforward. Pure gradient systems tend to behave like analog networks: they diffuse, equilibrate, and smear distinctions unless actively maintained. That can be useful for neuromorphic dynamics, but it is hostile to robust logic. Logic wants crisp separations—high versus low, on versus off—and it wants those separations to be regenerated repeatedly in the face of noise and leakage.

CMOS achieves this with complementary devices: an n-type pull-down and a p-type pull-up, arranged so that one is off when the other is on. The result is a gate whose output is actively driven to a rail, not passively left to drift. The chemiosmotic analog seeks the same restoration property by building a push–pull gradient system. Instead of electrons and holes, the complementary carriers are effectively acidifying and basifying fluxes—H+ and OH- (or an equivalent pair of complementary ionic processes). When one network is engaged, it drives the output toward an "acid rail." When the other network is engaged, it drives toward a "base rail." Either way, the output is actively restored to a known domain.

Noise margin is the first payoff. In chemiosmotic substrates, absolute values drift with hydration, temperature, and slow chemistry. A complementary scheme defines rails by *relative* chemical potential domains rather than absolute pH numbers, and it measures outputs differentially: "more acidic than reference" versus "more basic than reference." This makes the system resilient to common-mode drift. It is the chemical analog of common-mode rejection in differential amplifiers.

Static dissipation is the second payoff. In non-complementary protonic logic, maintaining a state often requires continuous pumping, because leakage and diffusion erase the distinction. A complementary gate can be designed so that, once an output has been driven to a rail and the controlling pathways close, there is minimal steady flux—only leakage. The gate's job becomes to switch occasionally and then hold. This does not eliminate the need for refresh, but it changes refresh from "constant fight against diffusion" to "periodic correction for leakage," which is a qualitative improvement.

Restoration is the third payoff and the most important one. Without restoration, cascaded gates degrade. With restoration, you can build depth: outputs from one stage can drive the next without accumulating analog blur until meaning is lost. Complementary logic is therefore not merely an implementation trick; it is the architectural condition for building large-scale protonic logic networks that behave predictably.

5.2 Carrier symmetry and neutrality: how to avoid "acid accumulation" failure

The central hazard of proton-only computation is that it tends to "run away" chemically. If you repeatedly pump protons into one region (or out of another) without an equal and opposite

compensation, you do not just create information—you create chemical damage. You change pH beyond useful ranges, you alter membrane and surface chemistry, you induce corrosion or denaturation (in bio-derived components), and you eventually saturate buffers so that state becomes pinned and unresponsive. This is the "acid accumulation" failure mode: the chip becomes chemically biased in one direction until it can no longer compute.

Complementary H+/OH- logic is a strategy for neutrality. The guiding principle is not philosophical symmetry; it is charge and mass balance. Any sustained computation must manage where protons come from and where they go. If the system draws protons from water dissociation, it must also manage hydroxide. If it moves protons from compartment A to B, it must eventually move them back or export them to a reservoir. If it creates local acidity to represent a "1," it must create local basicity elsewhere or provide a neutralization pathway, otherwise the substrate chemically drifts until it fails.

There are multiple ways to implement neutrality, and they do not all require literal OH- transport as a "carrier." In some designs, the complementary action is provided by counterion motion plus buffering: one path effectively acidifies, the other effectively de-acidifies by removing protons or supplying buffering base. In other designs, OH- (or equivalent base carriers) is explicit, and the system truly has two complementary ionic networks that can be driven in opposite directions.

In all cases, the same system-level constraint applies: the chip must have a neutrality budget. It must know, at least statistically, how much net acidity it has injected into each domain over time and how it will be compensated. This naturally suggests a role for reservoirs: regions that can absorb or supply protons and bases without being harmed—chemical ground and chemical Vdd, in effect. It also suggests that long computations are possible only if the system includes reset/restore operations that periodically rebalance chemistry, just as a digital system includes garbage collection or refresh.

Neutrality is also intimately tied to materials. Many substrates are not chemically symmetric: they tolerate acidic excursions better than basic ones, or vice versa. Therefore "carrier symmetry" is more a conceptual discipline than a guarantee of identical physics. The point is not to force perfect symmetry but to avoid unbounded drift. Complementary logic is the method by which a chemiosmotic chip becomes a sustainable engine rather than a one-way chemical experiment.

5.3 Gate primitives: push–pull chemiosmotic switching and differential thresholds

A "gate" in complementary protonic logic is best understood as a controlled competition between two flux pathways: an acidifying pathway and a basifying pathway. The output node is a compartment (or an interfacial region) whose state can be shifted toward one rail or the other

depending on which pathway is enabled. The simplest primitive is therefore push–pull switching: one side drives the output in one chemical direction, the other side drives it in the opposite direction.

In practice, the gate must do three things: select, restore, and isolate. Select means that inputs (which may be chemical, electrical, or optical) determine which pathway is open. Restore means that whichever pathway is selected drives the output decisively toward a rail rather than leaving it in an ambiguous mid-state. Isolate means that once the output reaches its rail, conductance pathways close so that static dissipation is minimized.

Differential thresholds are the key to robustness. Instead of representing logic states as absolute pH values, you represent them as a difference relative to a reference. This reference might be a paired compartment that tracks common-mode drift, or it might be a local buffered baseline. A threshold element—implemented as a pH-sensitive channel, a field-effect sensor with hysteresis, or a chemical switch—decides whether the output has crossed into the "acid domain" or "base domain." By building gates around differential thresholds, the architecture becomes tolerant to slow baseline shifts and manufacturing variability. The gate does not ask "is pH = 6.8?" It asks "is this region more acidic than its companion by more than X?"

Gate families can then be constructed analogously to CMOS: inverters, NAND-like and NOR-like structures, and transmission gates (controlled valves that pass or block chemical potential). The difference is that the outputs are not voltages; they are localized chemical potentials. Yet the combinational logic idea can still apply if restoration exists. A protonic inverter, for example, is a device where an input that tends to acidify causes the output to basify, and vice versa, implemented through mutually exclusive pathway control.

An important nuance is that chemiosmotic gates can naturally incorporate time. Because gradients build and relax with characteristic time constants, the gate is inherently a dynamical element. This creates both opportunity and complexity. Opportunity, because timing can be encoded in the physics—delays, integration, and refractory periods come for free. Complexity, because logic no longer occurs in instantaneous levels but in trajectories. Complementary design can mitigate this by making restoration fast relative to leakage and by defining clear "evaluation windows," analogous to dynamic logic in electronics but with chemical refresh.

Finally, rectification reappears as a gate design feature. A useful gate often needs to allow a strong drive toward a rail but prevent reverse flow that would undo state. Rectifying channels and asymmetric membranes can give directionality to the push–pull paths, ensuring that switching is decisive and holding is stable.

5.4 Candidate material stacks: hydrated polymers, oxides, and selective membranes

The feasibility of complementary protonic logic lives in the material stack. The stack must support proton conduction (and possibly hydroxide conduction), stable gating, low leakage when “off,” and repeatability under cycling. Three material families dominate the plausible design space.

Hydrated polymers are attractive because they can support significant proton mobility when properly hydrated, can be patterned, and can host functional groups that provide buffering and pH sensitivity. They also allow embedded reservoirs and graded conduction regions. Their challenges are drift, swelling, and environmental dependence: humidity and water content can change conductance dramatically. In complementary logic, this is manageable only if the architecture is differential and includes calibration and refresh, and if packaging controls hydration as a design parameter rather than an afterthought.

Oxides and inorganic proton conductors offer a different path. Some oxides can conduct protons under certain conditions and can be integrated with microfabrication. Oxides often provide better mechanical stability and pattern fidelity than soft polymers. They can also host interfacial gating phenomena and can be engineered with layered structures that control ionic selectivity. Their challenges are that proton conduction may require specific hydration or temperature regimes, and that chemical stability must be ensured under repeated acid/base cycling. Complementary logic imposes a harsh requirement: the material must survive oscillation between domains without gradually changing its own chemistry in a way that alters the device.

Selective membranes are the third family and are essential regardless of conductor choice. The membrane determines selectivity, prevents uncontrolled counterion flux, and provides the capacitive boundary for delta_psi. For complementary logic, membranes must also support controlled gating and perhaps different selectivities for the two networks (acidifying versus basifying). This suggests multilayer membranes: one layer for mechanical integrity, one for selective transport, one for functionalized gating sites. It also suggests the use of composite stacks: polymer + oxide + functional monolayer, each contributing a different property.

A practical early path is likely hybrid: use materials that are excellent protonic media (hydrated polymers or tailored oxides) for the ionic network, and use electronic devices nearby for readout and stabilization. The point is not to smuggle CMOS back into the core, but to let electronics provide precision metrology while the protonic substrate provides statefulness and restoration physics.

Ultimately, complementary protonic logic is an attempt to make chemiosmotic computation scalable in depth, not just in width. Battery arrays scale in number of nodes. Complementary logic aims to scale in the ability to build long chains of operations without losing meaning. That

demands a material stack that can be driven repeatedly, restored reliably, and kept chemically neutral over time. The architecture is therefore not merely “protonics with a plus sign.” It is a disciplined attempt to replicate the deepest lesson of CMOS—complementarity and restoration—using the language of gradients, membranes, and controlled chemical domains.

6. Architecture III: Protonic FPGA via Reconfigurable Canal Networks

6.1 Channels as wires: dispersion, crosstalk, and spatial computing constraints

A protonic FPGA is the attempt to make chemiosmotic systems *programmable at the routing level*, not just at the device level. The core substrate is a network of channels—nanochannels, microchannels, or membrane-defined conduits—through which protonic signals and gradients can propagate. In this architecture, channels are the analog of wires. But protonic “wires” are not metal traces carrying nearly lossless electromagnetic signals. They are hydrated, chemically active pathways that carry state by diffusion, drift, and local re-equilibration. That difference forces an entirely different design intuition.

Dispersion is the first constraint. In a protonic channel, a sharp change at the source tends to spread as it travels. A brief acidification pulse becomes a wider, lower-amplitude waveform downstream. This is not a defect to be ignored; it is the channel’s native transfer function. Channels act like low-pass filters whose cutoff depends on length, cross-section, hydration, and surface chemistry. Therefore, “wire length” is not just a delay; it is a distortion. Long channels do not merely arrive later; they arrive smeared.

Crosstalk is the second constraint. Protonic channels live in an environment where fields and chemical potentials leak into each other. If two channels are spatially close or share a boundary medium, gradients can couple through diffusion or through electrostatic interactions at surfaces. In electronics, crosstalk is mainly capacitive and can be managed with shielding and spacing. In protonic networks, crosstalk can be chemical: local pH changes can alter the conduction properties of neighboring channels, and shared reservoirs can create hidden coupling pathways. This is dangerous for logic but potentially useful for analog spatial computation, where coupling is a feature. The key is to decide intentionally which regime you’re building: isolated wires for routed logic, or coupled fields for reservoir-like dynamics.

Spatial computing constraints define the third principle: in protonic networks, computation is inescapably geometric. You cannot pretend signals are identical regardless of position. Geometry determines attenuation, dispersion, coupling, and stability. As a result, the design vocabulary shifts from “netlists” to “maps.” A protonic FPGA is best thought of as a reconfigurable canal graph embedded in space, where distances, junction angles, and boundary conditions matter. This pushes the paradigm toward tasks that naturally live in spatial analog

form—pattern formation, signal integration, gradient-based inference, and dynamical filtering—rather than tasks that demand crisp symbol manipulation.

This is the conceptual reframe: channels as wires are not channels as perfect conduits. They are channels as controlled dispersive media. The “wire model” is more like transmission in a lossy, diffusive line than in a metallic conductor. Once accepted, this yields a new toolkit: design pulses to survive dispersion, use differential pairs to reject drift, place buffers to regenerate amplitude, and use geometry as a computational degree of freedom rather than an inconvenience.

6.2 Switch fabric: voltage-gated, light-gated, and chemically programmed routing

An FPGA is defined by its switch fabric: a reconfigurable interconnect that can be set to create many different circuits from the same substrate. In a protonic FPGA, the switch fabric is a set of controllable junctions and valves that open or close pathways between channels and compartments. These switches must be reconfigurable repeatedly, must be reasonably isolating when “off,” and must not destroy the gradients they route.

There are three principal control modalities: voltage-gated, light-gated, and chemically programmed.

Voltage-gated switches use electric fields to modulate channel conductance. In hydrated media and membrane systems, electric fields can change surface charge, alter local ion distributions, and gate channels in ways reminiscent of ion-gated transistors. The appeal is direct compatibility with electronic control: a CMOS periphery can program the protonic fabric by applying voltages to gates without physically opening the system. The risk is leakage and unintended coupling: electrodes can become shunt paths, and gating fields can perturb nearby channels, creating a control-crosstalk problem. Voltage gating is likely the first practical approach for fine-grained routing because it is addressable and fast, but it demands careful isolation strategies.

Light-gated switches use optical control to change permeability or to activate pumps that alter local pH in a way that opens or closes a valve. Light gating has the same advantage it had in gradient writing: it avoids conductive feedthroughs that compromise sealing. It also scales naturally to arrays, because light can address many junctions in parallel via patterned illumination. Light gating further allows wavelength multiplexing: different switches can respond to different colors, enabling orthogonal control layers. Its main challenge is photostability and cross-illumination: keeping the system from being inadvertently reprogrammed by scattered light.

Chemically programmed routing is the most radical and, in some sense, the most “FPGA-like” in an analog substrate. Here, routing configuration is stored as chemical state: a polymer region

that is protonated becomes permeable; a region that is deprotonated becomes impermeable; a gel swells to block a channel; a catalytic patch changes local hydration and conductance. Chemical programming can be nonvolatile or semi-volatile, meaning the routing fabric itself stores the configuration without continuous power. But chemical programming also invites the hard problems of drift and endurance. If you rely on chemistry as configuration memory, you must prevent accidental reconfiguration by noise and must provide reset mechanisms.

In practice, an early protonic FPGA will likely be hybrid: a voltage-gated switch fabric for rapid and precise reconfiguration, augmented by light-gated or chemically programmed elements for coarse-grained mode switching and nonvolatile configuration storage. The essential concept is not which modality wins, but that the fabric is truly reconfigurable—capable of rerouting around defects, of changing coupling topologies for different tasks, and of shifting between "logic-like" and "field-like" computational modes.

6.3 Buffers as capacitors: weak acids/bases, reservoirs, and time constants

In electronics, capacitors store charge and stabilize rails; they also implement timing. In chemiosmotic systems, the analog of capacitance is buffering: the ability of a region to absorb or release protons (or bases) without large changes in pH. Buffering turns proton count into a smooth state variable and sets the time constants of response and decay. Therefore, buffers are not optional auxiliaries in a protonic FPGA; they are the essential elements that make routing and computation stable.

Weak acids and bases provide local buffering capacity. Placed at strategic nodes, they act like capacitors at junctions: they absorb sudden proton influx, preventing spikes from propagating as destructive excursions, and they release protons slowly, smoothing outputs. This makes channels behave less like raw diffusers and more like engineered signal lines. Buffers also provide noise tolerance: stochastic proton events are averaged over a chemical reservoir, reducing sensitivity to microscopic fluctuations.

Reservoirs play the role of larger capacitors and power supplies. A reservoir can be a dedicated compartment with high buffering capacity that serves as a reference or as a sink/source for rebalancing. In routing terms, reservoirs can clamp pH in regions, enabling stable reference levels against which differential signals are measured. In maintenance terms, reservoirs enable restore operations: if a region has drifted too acidic, connect it briefly to a base reservoir; if too basic, connect to an acid reservoir. The reservoir network becomes the backbone of housekeeping, but it can also become part of computation by providing bias fields.

Time constants are the computational gold. The effective RC constant of a protonic node is set by channel conductance (R) and buffer capacity (C). By choosing buffer chemistry and placement, you can engineer a spectrum of time constants across the fabric: fast nodes for

switching and event detection, slow nodes for integration and memory. This heterogeneity is difficult to achieve cleanly in electronics without explicit capacitors and careful leakage control; in protonics, it is native and programmable.

However, buffering introduces tradeoffs. Strong buffers increase stability but reduce sensitivity: more protons are required to move pH. Weak buffers increase sensitivity but amplify noise and make the system easier to saturate. Therefore, a protonic FPGA is likely to use hierarchical buffering: strong buffers at reference rails and major junctions, weaker buffers at local computational nodes, and specialized “integrator” nodes designed to accumulate small signals over long times.

The deeper insight is that buffers are not passive background chemistry. They are explicit circuit elements. Designing a protonic FPGA means designing a distributed buffer network that defines the analog storage and timing behavior of the entire substrate.

6.4 Compilation concept: mapping tasks onto canal graphs and scheduling restoration

An FPGA is useful only if there is a way to compile tasks into configurations. In protonic networks, compilation cannot be purely symbolic. It must respect geometry, dispersion, buffering, and drift. Therefore, the compilation concept is closer to mapping a dynamical system onto a physical medium than to placing logic gates in a grid.

Mapping begins with a canal graph: nodes (compartments, junctions, reservoirs) connected by channels (wires) with known conductances and coupling properties. Each switch in the fabric chooses whether edges exist and, in some designs, how strongly they conduct. Each node has buffer capacity and leak rate. This defines a dynamical network. To compile a task, you choose a subgraph and configure switches such that the network’s natural dynamics implement the desired computation.

For logic-like tasks, compilation will emphasize restoration and isolation. It will choose short channels, strong buffering at rails, differential encoding, and periodic refresh. It will avoid long diffusive paths that smear signals and will use regenerative nodes (threshold elements) to clean up waveforms. The compiled design will resemble a staged pipeline with local resets, more like dynamic logic than static logic.

For analog field-like tasks, compilation will embrace coupling. It will choose long channels, controlled diffusion, and distributed buffers that create desired time constants and spatial smoothing. It will use the network as a reservoir: inputs perturb the system, and readout nodes measure the resulting spatiotemporal patterns. In this regime, compilation is less about exact correctness and more about reproducible response classes: stable attractors, separable trajectories, and robust features.

Scheduling restoration is inseparable from compilation. In CMOS, you can often ignore maintenance during computation. In protonic FPGAs, you must schedule refresh, recalibration, and neutrality correction as part of the program. The compiler must know leak budgets and must allocate time to re-zero baselines, clamp rails, and quarantine faulty nodes. This resembles real-time operating systems more than traditional compilers. The "program" is a schedule of configurations and maintenance actions: configure, drive, measure, restore, reconfigure.

An advanced compilation model would include adaptive calibration loops. Because each physical device will differ slightly, the system can measure its own transfer functions and adjust configuration accordingly. In other words, the protonic FPGA could be self-characterizing: it learns its own canal graph parameters and compiles tasks onto the substrate it actually is, not the ideal substrate it was supposed to be. This is a natural fit for chemiosmotic systems, where variability is the rule. Rather than fight it, compilation can harness it by treating the chip as an individually calibrated instrument.

In summary, a protonic FPGA via reconfigurable canal networks is the architecture that makes chemiosmotic computation programmable in the deepest sense. It accepts that protonic media are spatial, dispersive, and stateful, and it uses reconfigurable routing, buffering-as-capacitance, and explicit maintenance scheduling to turn those constraints into a computational fabric. This is the point where "chemiosmotic chip" stops being a single device or a single array and becomes a platform: a reconfigurable medium whose computation is the controlled shaping of gradients in space and time.

7. Architecture IV: Gradient-Native Neuromorphic Chips

7.1 Neurons as thresholds in chemical fields: integrate-and-fire in delta_pH space

Neuromorphic computing is often described as "brain-like," but the deeper commonality is not anatomy. It is physics: computation carried by slow, stateful variables that integrate, leak, and occasionally switch nonlinearly. Chemiosmotic substrates are naturally in this class. They provide continuous-valued state (delta_pH, delta_psi, hydration, protonation) that evolves over time and can cross thresholds that trigger sharp events. A gradient-native neuromorphic chip therefore does not need to imitate spiking neurons by building elaborate electronic circuits; it can realize integrate-and-fire behavior directly in chemical field space.

The basic neuron primitive is a compartment or interfacial region with (i) a storage capacity for protons (buffer capacity), (ii) a leakage pathway that causes relaxation (a "leak conductance"), and (iii) a threshold mechanism that triggers a rapid discharge or reconfiguration when delta_pH or pmf crosses a boundary. Integration occurs when upstream events deliver small increments of acidity or basicity—through gated channels, diffusive coupling, or pump-driven

writes. Leak occurs as protons diffuse away, recombine, or are neutralized by buffering and slow exchange with a reservoir. The neuron's "membrane potential" is literally a membrane potential and/or a pH offset. When the state crosses threshold, a gated channel opens, a rectifying pathway engages, or a valve changes permeability, producing a burst-like event: rapid proton flux that alters downstream nodes.

In this view, spiking is not a digital artifact; it is an energetic discharge event. A spike can be defined as a transient, high-flux state that is easy to detect electronically or optically. Importantly, the spike does not require perfect reproducibility at the single-event level. Neuromorphic computation tolerates variability as long as spike statistics and coupling patterns remain stable.

The time constants are tunable and meaningful. In electronics, time constants must often be engineered with explicit capacitors and resistors. In gradient-native systems, they come from geometry and chemistry: buffer capacity, channel length, and membrane leakage. This makes a gradient-native neuron intrinsically multi-timescale. It can have a fast response component (delta_psi changes across a capacitive membrane) and a slow integrative component (delta_pH evolving in buffered volume). That dual timescale is valuable: it allows rapid event triggering on top of slow state accumulation, producing rich dynamical behavior without additional circuitry.

Finally, "neuron" here is not a single fixed device. It is a role in a network. Any region that can integrate input, leak over time, and generate a thresholded event can serve as a neuron. Gradient-native neuromorphic chips therefore lean toward heterogeneous neurons: some fast and excitable, some slow and integrating, some with refractory behavior due to slow chemical recovery. This heterogeneity is not an inconvenience; it is the substrate's natural computational repertoire.

7.2 Synapses as buffering capacity: weight storage as local chemistry

If neurons are threshold elements, synapses are the adjustable couplings between them. In a chemiosmotic substrate, the most natural way to store a synaptic "weight" is not as a digital number or even as a stable conductance alone, but as a local chemical property that shapes flux: buffering capacity, local permeability, or protonation state of a functionalized region. Among these, buffering capacity is particularly powerful because it defines how strongly a given input event changes the downstream state.

A synapse can be implemented as a junction region between channels or compartments where the local buffer chemistry is adjustable. A stronger buffer means that incoming proton flux produces a smaller change in delta_pH—equivalently, a smaller postsynaptic effect. A weaker buffer produces a larger delta_pH response—equivalently, a stronger synapse. In this way, weight is encoded as the "chemical capacitance" of the synaptic region. That makes the synapse

inherently analog and naturally bounded: it cannot grow without limit because buffer capacity has physical constraints.

Alternatively, weight can be encoded in permeability. A synaptic pathway may contain a gate whose baseline conductance can be modified by chemical state: protonation of a polymer, hydration of a channel wall, or conformational state of a membrane element. The effective conductance then determines how much gradient transfer occurs per event. Conductance-based weights and buffering-based weights are not mutually exclusive; they can be layered: conductance sets how much flux can pass, buffering sets how strongly that flux changes state.

This substrate makes synapses "local" in the most literal sense. In electronic systems, weights live in memory arrays that are logically local but physically distant from neurons unless careful co-location is engineered. In a chemiosmotic neuromorphic chip, weights can live exactly where coupling occurs: the synapse is the place where chemistry is configured. That co-location collapses the distinction between compute and memory. It also makes learning rules physically plausible, because the quantities that would drive learning—local delta_pH excursions, local flux events, local timing relationships—are present at the synapse itself.

Weight readout need not be explicit. In many neuromorphic paradigms, weights are not "read" as numbers; they are expressed as behavior. Here, the synapse's chemistry expresses itself as response amplitude and time constant. The system's output reflects weights without requiring separate memory access. This is one of the deepest advantages of gradient-native neuromorphic designs: the chip computes in the same medium that stores its parameters.

7.3 Metaplasticity and volatility: embracing drift as a computational feature

The instinct of conventional engineering is to suppress drift. Chemiosmotic neuromorphic chips can do something more interesting: they can use drift as a functional form of volatility and metaplasticity. In brains, synapses are not static; they change on multiple timescales. Some changes are short-lived (facilitation, depression), others persist (long-term potentiation/depression), and still others modify the synapse's ability to change (metaplasticity). A chemiosmotic substrate naturally supports this hierarchy because chemical state variables relax over time and can change their own dynamics.

Volatility is the simplest form: weights that decay unless reinforced. This is not a defect; it is a feature for continual learning in changing environments. A volatile synapse implements forgetting and prevents saturation. In chemical terms, volatility can be engineered by choosing buffer chemistries and gating mechanisms that relax toward baseline with a designed time constant. This yields short-term memory and soft persistence.

Metaplasticity is the next level: the synapse's susceptibility to change depends on its recent history. Chemiosmotic systems naturally exhibit this because local chemistry alters transport

properties. For example, repeated acidification may temporarily change channel permeability or buffer availability, making subsequent events more or less effective. This creates an adaptive learning rate. In neuromorphic learning, adaptive learning rates are valuable because they stabilize training and prevent runaway weight growth.

Drift can also serve as exploration. Many learning systems require some form of stochasticity or variability to explore configuration space. In chemical substrates, small fluctuations and slow drift can provide this "noise floor," enabling the system to sample nearby states. Coupled with a selection mechanism—such as reinforcement by repeated patterns—this becomes a physical analog of stochastic gradient descent: the system wanders, and successful trajectories get reinforced by local chemical changes.

Of course, embracing drift does not mean accepting unbounded chaos. The architecture must distinguish between useful volatility and destructive instability. The difference is whether drift is bounded by homeostasis and whether it is reversible. Useful volatility has a time constant and returns toward baseline. Destructive drift accumulates and moves baselines permanently until the system becomes biased or damaged. Gradient-native neuromorphic chips therefore require designed homeostatic loops: mechanisms that keep average pH within safe ranges, prevent depletion of buffering, and periodically restore neutrality. Within those bounds, drift is not the enemy; it is the substrate's way of providing continuous-time learning dynamics without expensive digital bookkeeping.

7.4 Learning rules: timing, propagation delays, and local homeostasis

Learning rules in neuromorphic systems describe how synapses change in response to activity. In gradient-native substrates, the most natural learning rules are local, timing-dependent, and homeostatically constrained—because those are precisely the signals available in the physics.

Timing enters through propagation delays in channels and relaxation times in compartments. When a neuron-like node fires (a threshold event), it produces a pulse that travels through channels and arrives at downstream synapses after a delay determined by geometry and medium properties. Meanwhile, the postsynaptic node is integrating its own inputs and may reach threshold earlier or later. The relative timing between pre- and post-events can be detected chemically: overlapping pH transients at a synapse create nonlinear interactions that do not occur if the events are separated in time. This provides a natural substrate for spike-timing-dependent plasticity (STDP)-like behavior, without needing an explicit clock. If pre and post events coincide, the synapse experiences a stronger excursion and changes more. If they are separated, the excursion is weaker and changes less or changes in the opposite direction.

Propagation delays therefore become computational resources. By shaping channel lengths and buffering, one can tune delay distributions across the network. This is a powerful idea: instead

of adding delay lines electronically, the chip's physical geometry provides a temporal embedding. This supports reservoir computing regimes where rich temporal dynamics create separable trajectories for different inputs.

Local homeostasis is the constraint that keeps learning sane. Without homeostasis, learning in a chemiosmotic substrate risks saturating pH extremes, exhausting buffers, or creating permanent biases that prevent further adaptation. Homeostasis can be implemented by slow feedback that stabilizes average pH, normalizes synaptic strengths, and prevents runaway excitation. In chemical terms, this might be a slow leak to a reservoir that clamps baseline, periodic refresh pulses that re-center distributions, or adaptive gating thresholds that shift when activity is too high or too low.

A particularly chemiosmotic-friendly learning pattern is "three-factor" learning: local activity plus a modulatory signal that biases plasticity. In biology, this is often dopamine-like reinforcement. In a chemiosmotic chip, a modulatory signal could be a global light pattern, a globally applied bias field, or a slowly varying chemical signal from a reservoir that indicates reward or error. The synapse changes only when local timing conditions are met *and* the modulatory condition is present. This enables reinforcement learning in a physically local way: global modulation sets when learning is allowed, local coincidences determine what changes.

Finally, because the substrate is analog and variable, learning rules should be expressed as robust tendencies rather than exact equations. The goal is not to guarantee identical weight updates every time; it is to guarantee that, over repeated experience, the system converges toward useful attractors. In this sense, gradient-native neuromorphic chips are less like deterministic digital processors and more like trained physical media. Their "program" is not a fixed set of instructions; it is a set of constraints, time constants, and homeostatic laws that shape how the medium adapts.

In summary, Architecture IV is where chemiosmotic computing looks least like CMOS and most like its own paradigm. Neurons are threshold events in chemical fields. Synapses are local chemistry and buffering. Drift is volatility and metaplasticity. Learning rules arise from timing overlaps and homeostasis rather than from centralized computation. This architecture does not compete by speed; it competes by being a native substrate for continuous-time, adaptive, stateful computation—precisely the domain where "chemistry as noise" becomes "chemistry as intelligence."

8. Architecture V: Hybrid Chemiosmotic System-on-Chip

8.1 Why hybrid wins early: electronics for timing, calibration, and I/O

A fully chemiosmotic chip is conceptually clean, but early successful systems are likely to be hybrid for the same reason early neural implants and lab-on-chip platforms are hybrid: the chemical domain is powerful but unruly, while electronics is precise but blind to chemical meaning without mediation. Hybrids win early because they let each substrate do what it is uniquely good at.

Electronics is unmatched at timing. Chemiosmotic dynamics are governed by diffusion, buffering, and leakage—processes with broad, device-specific time constants. That is not inherently bad, but it makes deterministic scheduling difficult. Electronics supplies the scaffolding: clocks, pulses, time windows, and synchronized addressing. Even if the chemiosmotic "core" is analog and continuous, an electronic periphery can define evaluation phases: write, settle, read, refresh. This brings a chemical system into a regime where experiments are repeatable and where computation can be staged.

Electronics is also unmatched at calibration. A chemiosmotic substrate will vary from node to node: membrane leakage, buffer capacity, pump orientation, sensor baselines, and channel conductance will not be identical. If you demand uniformity, you lose. Hybrid systems turn variability into a manageable parameter by measuring it and compensating. Electronics can map the array, estimate transfer functions, compute correction factors, and continuously re-estimate them as the system ages. In effect, the chip becomes a calibrated instrument. This parallels how modern image sensors and ADCs work: they are not perfect devices; they are devices paired with calibration logic that makes their imperfections predictable.

Finally, electronics is the natural interface for I/O. Most real systems—computers, networks, instruments—expect electrical signals. Even if the internal state is chemiosmotic, the system must deliver outputs to the world, accept inputs, and communicate. Electronics provides impedance matching to external systems, digitization when needed, data compression, and error checking. A hybrid architecture therefore solves the adoption problem: you can plug a chemiosmotic core into existing infrastructure without demanding that the world become chemiosmotic.

There is a deeper conceptual reason hybrid wins early: it allows chemiosmosis to be used where it adds value without requiring that it replace everything. The hybrid approach treats chemiosmosis as a specialized compute/storage fabric—good at statefulness, analog dynamics, and local coupling—embedded inside a conventional control and communication frame. This is analogous to how GPUs, TPUs, and analog accelerators are used today: not as replacements for CPUs, but as co-processors that dominate certain workloads.

8.2 The transducer layer: pH-sensitive FETs, redox bridges, capacitive couplers

The hybrid chip is defined by its transducer layer: the interface that converts chemiosmotic state into electrical observables and electrical control signals into chemiosmotic actions. If the chemiosmotic core is the “wet computation,” the transducer layer is the “translator” that makes it legible and controllable.

pH-sensitive FETs are the most direct readout concept. A pH-sensitive interface alters surface charge and therefore modulates a field-effect device. This allows delta_pH to be read as a change in threshold voltage or channel conductance. In a hybrid system, arrays of such sensors can sit under compartments or near channels, providing continuous monitoring of state. The advantage is non-destructive readout: if the FET senses surface potential rather than drawing ionic current, it can observe without discharging. The challenge is drift: pH-sensitive surfaces age, foul, and change their response. This is precisely where hybrid calibration logic becomes essential: the electronic layer continuously recalibrates sensor baselines against reference nodes.

Redox bridges provide a different kind of transduction: conversion between proton gradients and measurable electrode potentials via reversible redox couples. In such a scheme, chemiosmotic state influences local redox equilibrium, which is sensed electrically. Redox bridges can also work in reverse: an applied potential can bias local chemistry, injecting or withdrawing protons through electrochemical reactions. The appeal is that redox systems can provide amplification and can couple strongly to chemical state. The risk is that they may be more invasive: redox reactions can consume reagents, introduce byproducts, and create long-term drift unless carefully bounded and replenished. Therefore, redox bridges are powerful but require strict bookkeeping and restoration.

Capacitive couplers are the most conceptually “CMOS-like” interface for delta_psi. A membrane is a capacitor; delta_psi is a voltage across it. If you can couple to that voltage without providing a DC leakage path, you can measure delta_psi and even use it as a signal. Capacitive pickoff resembles floating-gate sensing: a nearby electrode senses changes in charge distribution through capacitance. The advantage is minimal chemical intrusion. The challenge is small signals and noise: the coupling capacitance may be tiny, and parasitic capacitances may dominate. Again, hybrid electronics helps: it provides low-noise amplification, differential measurement, and filtering.

In control direction, transducers must also act as actuators. The hybrid system needs ways to open gates, drive pumps, and clamp baselines. Actuation can be optical (micro-LED arrays), electrical (gate fields), or electrochemical (redox actuation). The transducer layer is therefore

not a single device type; it is a stack: sensors, actuators, and coupling structures, all designed to preserve the chemiosmotic core's integrity while making it programmable.

A mature hybrid design will likely adopt differential, redundant sensing. It will pair each computational node with reference nodes and will read differences rather than absolutes. It will also embed self-test modes: the electronics can inject known perturbations and measure responses to detect fouling, leakage increases, and sensor drift. In other words, the transducer layer is not a passive window. It is a continuously operating measurement-and-control system.

8.3 Refresh as a design principle: chemical DRAM and periodic re-zeroing

Hybrid chemiosmotic SoCs will live or die by how they handle refresh. In pure electronics, DRAM is successful because refresh is systematic and affordable: charge leaks, so you periodically restore it. Chemiosmotic systems leak and drift in richer ways: membranes leak, buffers saturate, surfaces foul, and baselines wander. If you pretend these don't exist, the system will fail. If you build refresh into the operating model, the system becomes viable.

The useful analogy is chemical DRAM. A node's state—its pmf partition and local chemistry—decays with time. Refresh means recharging gradients and re-centering baselines before the stored information becomes ambiguous. But chemiosmotic refresh also includes neutrality management: you must prevent net acidification or basification over long times. Therefore refresh is not merely "recharge." It is "recharge plus rebalance."

Periodic re-zeroing is the most important refresh primitive. The system needs a known reference state. That might be a neutral baseline, or it might be a standardized bias state from which computation proceeds. The hybrid controller periodically connects nodes to reservoirs, opens channels to equilibrate, or applies reverse pumping to bring nodes back into range. This is comparable to offset correction in analog circuits: you periodically measure baseline and subtract it out. But here, re-zeroing is physical: you are moving matter and charge to restore a baseline state, not merely subtracting a number.

Refresh also provides fault management. Nodes will degrade at different rates. Some will leak too fast, others will become insensitive due to buffer saturation or fouling. In a hybrid SoC, the electronic layer can map node health and adaptively choose refresh rates per node—fast refresh for leaky nodes, slow refresh for stable nodes. It can also reassign roles: use leaky nodes for fast transient dynamics and stable nodes for memory. This turns variability into architecture.

Refresh is not free. It costs energy (photons, electrical bias, or chemical consumption) and time (during which computation may pause or be perturbed). Therefore, an advanced hybrid system will schedule computation around refresh, much like DRAM controllers schedule memory accesses. In neuromorphic modes, refresh may be integrated continuously as homeostasis:

small corrections applied constantly rather than disruptive periodic resets. In logic-like modes, refresh may occur in explicit phases, with computation happening in clean evaluation windows.

The deeper point is that refresh is not a patch. It is the enabling principle that turns a leaky chemical substrate into a reliable computational medium. Without refresh, chemiosmotic chips remain laboratory demonstrations. With refresh, they become systems.

8.4 Security and integrity: preventing contamination, spoofing, and state hijacking

A chemiosmotic SoC introduces a new threat model. In electronics, threats are typically electrical: probing, glitching, injecting faults, or exploiting software vulnerabilities. In hybrid chemiosmotic systems, threats include chemistry itself: contamination, spoofed signals, and state hijacking through the environment. If chemiosmotic chips ever become practical, their integrity will depend on treating chemical pathways as attack surfaces.

Contamination is the simplest and most dangerous risk. Because the substrate depends on hydration and ion transport, trace contaminants—ions, organics, surfactants, reactive gases—can change conductance, alter selectivity, foul membranes, and shift sensor baselines. In an open system, an attacker could introduce contaminants deliberately; in a closed system, contaminants may accumulate slowly from materials outgassing or seal degradation. Therefore, integrity requires chemical hygiene: controlled reservoirs, filtration, sealing, and continuous monitoring of baseline parameters. A hybrid controller should treat unexpected baseline drift as a potential fault or intrusion, not merely as noise.

Spoofing is the second risk. If outputs are based on pH-sensitive sensors, an external chemical or electrical perturbation might mimic an internal computational state. For example, stray fields could shift sensor readouts; temperature changes could alter pH and conductivity; light could inadvertently activate pumps or photoreactive materials. Therefore, secure designs will use differential sensing, multi-modal verification (pH plus delta_psi plus impedance signatures), and challenge–response self-tests: the chip applies known perturbations and checks whether responses match the internal state model. If they do not, the system flags integrity failure.

State hijacking is the third risk and is unique to substrates with persistent analog state. Because chemiosmotic nodes can store state in chemical form, an attacker might attempt to bias the system slowly, pushing it toward a chemically “captured” regime where outputs are predictable or learning rules are corrupted. This resembles data poisoning in machine learning but enacted physically. Defenses include periodic re-zeroing to erase accumulated bias, constrained operating ranges to prevent extreme drift, and cryptographic or procedural controls in the electronic layer that govern when learning is allowed. In three-factor learning terms, you can require an authenticated modulatory signal before plasticity occurs, preventing an attacker from simply feeding inputs that reshape the system.

Integrity also intersects with safety and reliability. A chemiosmotic chip can fail by chemical damage: pH extremes, corrosion, membrane rupture, gas generation in electrochemical actuation, or biofouling. Secure operation therefore includes safe operating envelopes enforced by the controller: clamp pH, limit cumulative acid/base exposure, detect rapid leakage changes, and quarantine compromised domains.

The hybrid architecture gives you the tools to do this. Electronics can run integrity monitors continuously, log state histories, and enforce policies: allowable illumination patterns, allowable gating sequences, allowable reservoir exchanges. It can implement redundancy and voting across multiple sensor channels. It can detect anomalies and shut down or reinitialize domains. In other words, the hybrid SoC can treat its chemiosmotic core as a managed, monitored environment—much as a modern system treats memory and peripherals as monitored resources with error detection and correction.

Hybrid chemiosmotic chips will therefore not merely be "bio-inspired accelerators." They will be systems that require a new discipline: chemical state management with auditability. If that discipline is embraced—timing control, calibration, refresh, and integrity monitoring—then chemiosmosis can move from a fragile physics demonstration to a robust on-chip computational fabric. The path to "advanced" chemiosmotic computing almost certainly runs through this hybrid stage, because it offers a practical way to keep the chemical substrate honest while letting it do the kinds of computation electronics does poorly: continuous-time adaptation, local statefulness, and gradient-native dynamics.

9. Manufacturing and Integration Pathways

9.1 CMOS compatibility vs bio-MEMS stacks: what "compatible" should mean

"CMOS-compatible" is one of those phrases that can mean almost anything, so a chemiosmotic chip program has to define it with discipline. In the strictest sense, CMOS compatibility means that the entire process can be executed within standard semiconductor fabs: temperature budgets, materials lists, contamination controls, and tool flows that do not poison transistor yield. Under that definition, most membrane- and hydration-dependent systems are not compatible, because they require soft matter, aqueous processing, and materials (ions, polymers, biological components) that fabs treat as contaminants.

In a more useful sense for this field, compatibility should be framed as *stack compatibility*: the ability to fabricate CMOS electronics normally, then add a post-CMOS "chemiosmotic stack" in a controlled, isolated packaging and assembly step. This reframes the question from "can a membrane go through a fab?" to "can a membrane be integrated above a finished chip without ruining it?" That is the realistic bridge. It mirrors how image sensors, MEMS microphones,

microfluidic cartridges, and biochips are often manufactured: the electronics are made using standard flows, and specialized functional layers are added later.

A bio-MEMS stack is therefore not the opposite of CMOS; it is often the path that preserves CMOS while allowing wet, ionic structures to exist in a segregated layer. The key requirement becomes: strong isolation between the wet chemiosmotic domain and the dry electronic domain, with well-defined transduction interfaces (capacitive, field-effect, optical) that minimize leakage. Under this definition, “compatible” means the following: (i) the electronic substrate remains electrically reliable and protected from ionic contamination, (ii) the chemiosmotic layer can be assembled reproducibly, and (iii) the two layers can communicate through defined, testable interfaces.

This is also where packaging becomes central. For chemiosmotic chips, packaging is not a box around a device; it is a functional requirement that maintains hydration, prevents contamination, and provides optical and fluidic access if needed. It may resemble a microfluidic package more than a plastic IC package. Therefore, in this field, “compatibility” should include package-level manufacturability: can the stack be sealed, filled, sterilized (if biological components are used), and kept stable over relevant lifetimes? If not, it is not truly compatible with deployment even if it can be fabricated once.

A pragmatic definition, then, is: CMOS-compatible if the chemiosmotic system can be integrated as a post-CMOS module with no exotic fab contamination risk, with yield that can be measured at wafer scale, and with packaging that stabilizes hydration and chemistry over time. This is the definition that moves the conversation from slogans to engineering.

9.2 Membrane placement and orientation at scale: templating, self-assembly, lithography

Membrane integration is the signature manufacturing problem because membranes are simultaneously delicate and decisive. They define selectivity, pmf retention, and the boundary capacitance that makes the whole paradigm work. A working single membrane device is easy compared to working arrays. Scaling requires that membranes be placed, aligned, and oriented across thousands or millions of nodes with tolerable defect rates.

Placement has three broad routes. Templating uses patterned surfaces that encourage membranes or membrane patches to land only where desired. This can be done through chemical functionalization (hydrophilic/hydrophobic patterning, charge patterning) or through topography (micro-wells, apertures, docking sites). The promise is that the surface becomes an “address grid” that catches membranes in predictable spots.

Self-assembly leans into the fact that bilayers and vesicles form naturally. The challenge is to connect that natural closure to engineered addressability. One approach is to form vesicles in situ in patterned cavities. Another is to deposit vesicles and induce fusion to create supported

bilayers that span apertures. A third is to use membrane patches (for example, protein-rich membrane sheets) and assemble them onto designed supports. Self-assembly is appealing because it can produce high-quality membranes at scale without micromanipulation, but it tends to produce distributions—size distributions, orientation distributions, defect distributions—that must be accepted and managed.

Lithography enters not as “pattern a membrane with UV light” in the usual semiconductor sense, but as patterning the supporting structures around which membranes assemble. Lithography defines apertures, cavities, rails, and coupling electrodes. It defines the geometry that controls dispersion, leakage, and capacitance. In other words, lithography sets the physical graph on which membranes become functional boundaries. The more the architecture relies on lithographic definition of compartments and routing, the more it can inherit semiconductor-style reproducibility.

Orientation is the subtler scaling problem. For light-driven pumps and channels, the directionality matters. Random orientation produces cancellation and noise. At small scale, one can use biochemical tricks to orient proteins: tags that bind to specific surfaces, asymmetric lipid environments, or electric-field alignment. At large scale, one must choose an orientation strategy that is robust and manufacturable. This may involve asymmetric anchoring sites, patterned charges, or polarized illumination protocols that bias insertion direction. The key is to treat orientation as a yield parameter. A membrane is not “good” if it merely exists; it is good if its functional components are oriented correctly enough that net pumping and gating behave deterministically.

Finally, there is the question of heterogeneity tolerance. A mature architecture may accept that not all nodes have perfect orientation and instead incorporate mapping and redundancy: identify nodes that pump strongly versus weakly, assign them roles accordingly, and route around poor performers. This is conceptually similar to how memory arrays deal with defective cells. It implies that membrane placement and orientation at scale is not only a materials problem but also a system design problem: manufacturing variability becomes a parameter the architecture expects and manages.

9.3 Sealing and leak testing: metrology for pmf retention and channel defects

Sealing is the quiet killer. A chemiosmotic chip can appear to work in the moment—pumps produce a measurable pH shift—but fail as an energy storage and computing substrate because leakage erases state too quickly. Therefore, manufacturing must include metrology for pmf retention and defect detection, not merely functional demonstration.

Leak testing must operate at the scale of the relevant leakage paths. In electronic devices, you test for shorts, opens, leakage current, and dielectric integrity. In chemiosmotic devices, the

analogous tests include proton leakage, ion leakage, water permeability changes, and unintended shunt paths created by interfaces between membranes and solids.

A fundamental metric is retention time: how long a node holds a specified pmf (or a specified delta_pH and delta_psi partition) before it decays below a usable threshold. Retention is not a single number; it is a distribution across the array. Manufacturing must measure that distribution and sort devices, or at least classify nodes within a device. This is likely to involve accelerated tests: charge the node, measure decay, infer leakage parameters. In arrays, this can be automated: the electronic layer writes a pattern, reads it over time, and produces a retention map.

Defect localization is equally important. A leak might occur through a pinhole in a membrane, a faulty seal at an aperture edge, or a damaged channel wall that creates unintended coupling. The manufacturing flow should include tests that localize defects enough to allow quarantine. This encourages an architectural posture: treat the array as a fabric where some percentage of nodes will be defective, and design so that defects can be isolated and bypassed rather than causing catastrophic failure.

Metrology also needs to detect channel defects. In a reconfigurable canal network, channels are the wires. Their conductance distributions and crosstalk properties must be within expected ranges. A manufacturing test can probe channels by applying controlled perturbations and measuring downstream responses. This becomes a kind of system identification: the chip measures its own canal graph parameters. Such self-characterization is not a luxury; it is a realistic way to handle variability.

Crucially, leak testing must include chemical and environmental dimensions. Hydration level, temperature, and ionic composition affect leakage and transport. Therefore, a serious manufacturing flow will test under controlled environmental conditions and may include built-in sensors for humidity and temperature, so that performance can be normalized. The production “spec” will likely include both device parameters and operating environment requirements, because the substrate is intrinsically coupled to its chemical context.

9.4 Reliability physics: the protonic analogs of electromigration and dielectric breakdown

Reliability is where chemiosmotic chips will either become real technology or remain fragile prototypes. Electronics learned hard lessons about failure: electromigration in interconnects, dielectric breakdown in gate oxides, hot-carrier effects, corrosion, and packaging-induced stress. Protonic and chemiosmotic systems will have their own analogous failure physics—some familiar, some uniquely chemical.

The protonic analog of electromigration is not atoms drifting under electron wind, but *material and chemistry drifting under sustained ionic flux and pH gradients*. Repeated proton flow can

gradually change polymer hydration, rearrange functional groups, and alter channel surface charge. In membrane systems, sustained gradients can reorganize lipids or stress the membrane mechanically. In oxide systems, proton insertion and removal can change lattice structure or defect distributions. Over time, the very pathways that carry signal can change, shifting conductances and thresholds.

The analog of dielectric breakdown is membrane failure and seal failure. A membrane is a dielectric barrier separating ionic solutions. High delta_psi across a thin membrane can create strong electric fields that promote defect growth, pore formation, or catastrophic rupture. Even if rupture does not occur, small defects may enlarge under stress. In addition, electrochemical reactions at interfaces can generate gas or change local pH, accelerating damage. Therefore, safe operating envelopes must be defined not only for average pH but for instantaneous delta_psi and field stress.

Fouling and contamination are additional reliability mechanisms with no true analog in dry CMOS. Surfaces exposed to aqueous environments adsorb molecules, accumulate ions, and change wettability. Bio-derived components can denature or be attacked by microbes if sterile conditions are not maintained. Even purely synthetic systems can suffer from polymer aging, hydrolysis, and slow leaching of additives. Reliability engineering must therefore include chemical cleanliness, antimicrobial strategies if needed, and materials selection that resists long-term aqueous exposure.

Mechanical stress is also critical. Hydrated polymers swell and shrink. Membranes flex. Temperature cycles change expansion and may introduce microcracks at seals. Therefore, the packaging and substrate must be designed as a mechanical system, not merely an electrical one. Failure may occur at interfaces where soft matter meets hard matter—precisely the seals that also dominate leakage.

Finally, reliability must be treated as a system property rather than a device property. Because chemiosmotic chips will likely be heterogeneous and require refresh, calibration, and mapping, long-term reliability can be achieved even if individual nodes degrade—provided the system can detect degradation and reroute or reassign roles. This suggests that the protonic analog of ECC memory and fault-tolerant computing will be built in from the start: redundancy, reconfiguration, and health monitoring are not optional; they are the reliability strategy.

In summary, manufacturing and integration pathways for chemiosmotic chips are not “just fabrication.” They are the formation of a new discipline that sits between semiconductor processing, MEMS, microfluidics, and bioengineering. CMOS compatibility should be defined as stack compatibility and deployable packaging. Membrane placement and orientation must be engineered as yield parameters. Sealing and leak testing must become wafer-scale metrology.

Reliability must be reconceived in terms of chemical drift, membrane stress, fouling, and system-level fault tolerance. When these become explicit engineering practices rather than afterthoughts, chemiosmotic chips stop being a fascinating analogy and become a buildable technology class.

10. Performance Metrics and Benchmark Tasks

10.1 Core metrics: pmf magnitude, retention time, write energy, endurance, variability

A chemiosmotic chip cannot be evaluated with the metrics we reflexively use for CMOS, because the primitive is not switching speed; it is controllable, persistent gradient state. The core metrics therefore begin with how much usable proton-motive force (pmf) the device can create, how long it can hold it, how expensive it is to write, how many cycles it can survive, and how uniform it is across an array. These are the "device physics" numbers that determine whether any higher-level architecture is viable.

pmf magnitude is the headline metric: the maximum pmf the node can sustain under realistic operating conditions. But magnitude alone is not enough; the partition matters. A system that achieves a high delta_psi but collapses delta_pH under load may behave differently than one that builds stable delta_pH with modest delta_psi. Therefore pmf should be reported with its partition: how much is stored as delta_pH and how much as delta_psi, and how that partition changes under load.

Retention time is the true gatekeeper. It is the time for a node's state (pmf or a defined component) to decay below a defined usability threshold. Retention must be reported as a distribution across nodes, not as a best-case number, because array-level computation depends on the tails. Retention should also be measured under relevant operating conditions: hydration, temperature, and ionic composition. A retention time measured in an ideal buffer at constant temperature is not the same as retention in real packaging and duty cycles.

Write energy is the energy cost to change state by a defined amount. In optical writing, this includes photon energy and optical coupling losses; in electrical writing, it includes electrical energy and any electrochemical inefficiency; in hybrid writing, it may include both. Write energy should be normalized to the achieved state change: energy per delta_pH increment per node, or energy per pmf increment. This matters because a chemiosmotic system may be attractive precisely when it can operate with low average energy in continuous-time regimes, even if it is slow.

Endurance is the number of write/erase or charge/discharge cycles a node can undergo before its behavior changes unacceptably. Chemiosmotic nodes are vulnerable to fatigue mechanisms:

membrane damage, fouling, buffer depletion, and changes in channel permeability. Endurance must therefore be measured under cycling protocols that mimic intended use, including refresh. A node that can be charged once is not a device; a node that can be charged and refreshed millions of times with stable behavior is.

Variability is not merely a nuisance but a first-order design constraint. It includes node-to-node differences in pump strength, leakage, buffering capacity, and sensor calibration, as well as within-node drift over time. Variability should be measured and reported because it determines whether the architecture can rely on uniform behavior or must rely on mapping and adaptive control. In chemiosmotic systems, "performance" is often the performance of a calibrated ensemble, not of an idealized identical device.

The practical result is that any serious performance report should include: pmf magnitude with partition, retention distribution, write energy per state increment, endurance under cycling, and variability maps. These are the chemiosmotic analogs of Vdd, leakage, switching energy, lifetime, and mismatch in electronics.

10.2 System metrics: fan-out, routing loss, refresh overhead, and thermal coupling

Once the core device metrics exist, system-level metrics determine whether arrays can be built into useful computational fabrics. These metrics resemble electronic systems metrics in spirit but differ in what they mean physically.

Fan-out is the ability of one node's output to influence multiple downstream nodes without losing meaning. In electronics, fan-out is limited by drive strength and capacitive load. In chemiosmotic networks, fan-out is limited by gradient droop, diffusion dilution, and load coupling. If one node's gradient is used to drive many downstream couplings, each coupling draws from the same stored pmf and may reduce the source state. Therefore, fan-out must be defined as "how many downstream synapses can be driven while preserving the source state within tolerance," which in turn depends on buffering and the existence of restorative mechanisms.

Routing loss captures what happens as gradients move through channels. In protonic networks, routing loss is not just resistive; it includes dispersion, equilibration with intermediate buffers, and leakage into neighboring channels (crosstalk). Routing loss therefore measures how signal amplitude and separability degrade with distance and with the number of junctions. A useful way to characterize this is with transfer functions: input perturbation at node A, measured response at node B as a function of time and distance. For "logic-like" modes, you want low loss and fast restoration; for "reservoir-like" modes, controlled loss and mixing may be desirable. Either way, it must be measured.

Refresh overhead is the fraction of time and energy spent on maintenance rather than on "useful computation." This is the chemiosmotic analog of DRAM refresh cost, but potentially larger because maintenance includes neutrality control and recalibration. Refresh overhead can be expressed as: energy per second spent on refresh at steady operation, and time fraction devoted to refresh operations in a scheduled program. A chemiosmotic architecture that appears low-power but requires constant aggressive refresh may lose its advantage. Conversely, a system with modest refresh that runs continuously in event-driven mode may be compelling even if its peak performance is low.

Thermal coupling matters because chemiosmotic systems are temperature-sensitive. Diffusion coefficients, reaction kinetics, membrane permeability, and sensor baselines all vary with temperature. On-chip electronics generate heat, and optical writing injects heat. Therefore, thermal coupling is a systems metric: how temperature gradients across the chip correlate with state drift and error rates. Hybrid SoCs must treat thermal management as a computation integrity issue, not merely a reliability issue. A mature system will include thermal sensors and may include thermal-aware scheduling: avoid heavy optical writes near sensitive nodes, interleave refresh with cooling, or use differential designs that cancel common-mode thermal drift.

Together, these system metrics—fan-out, routing loss, refresh overhead, thermal coupling—determine whether a chemiosmotic chip can scale beyond a lab demonstration and behave as a stable computational fabric.

10.3 Benchmark tasks: sensing-to-inference loops, reservoir computing, adaptive control

Chemiosmotic chips should not be benchmarked by trying to imitate a CPU. They should be benchmarked on tasks where their native strengths matter: continuous-time statefulness, local coupling, and intimate interaction with chemical environments. Three benchmark classes align naturally with the architectures described earlier.

Sensing-to-inference loops are the first class. Here the chip is directly coupled to a chemical or physical sensor environment—gas composition, pH in microfluidics, ionic signatures, biochemical markers—and must transform sensory inputs into a classification or decision. Chemiosmotic substrates shine because sensing and computation can be co-located in the same medium: the sensor directly perturbs local chemical state, which then propagates through a reservoir or neural fabric. The benchmark is not "accuracy on ImageNet." It is performance on real chemical sensing tasks: robustness to drift, low power in continuous operation, and ability to adapt to sensor aging.

Reservoir computing is the second class. Protonic canal networks naturally implement reservoirs: coupled, dissipative dynamical systems with rich temporal response. In reservoir

computing, the internal network is not trained extensively; it is used as a dynamical expansion of inputs, and only the readout layer is trained. This fits chemiosmotic systems because the internal variability and analog dynamics are features, not bugs. A benchmark here could involve time-series prediction, pattern recognition in temporal signals, or classification of sensor streams. The key metrics are separability of trajectories, stability over time, and energy per inference in steady-state operation.

Adaptive control is the third class. Many control problems require slow integration, local memory, and continuous-time response: stabilizing a system under changing conditions, adjusting parameters as the environment drifts, and learning from feedback. Chemiosmotic neuromorphic chips can embed learning rules in local dynamics and can maintain state without constant high-frequency switching. Benchmark tasks could include controlling microfluidic flows, regulating chemical reactors at small scale, or performing adaptive filtering in sensor networks. The key metrics are stability, adaptation speed, and resilience to noise and parameter drift.

These benchmarks share an important property: they reward systems that are stateful and robust rather than systems that are fast. They also reward substrates that can be tightly coupled to the physical world without expensive transduction overhead. That is the regime where chemiosmotic computation can be meaningfully "better," not by replacing electronics, but by dominating a niche where electronics is awkward.

10.4 What would count as a decisive demo: three "signature" experiments

Because the field is easy to hype and hard to prove, it needs decisive demonstrations—experiments that, if achieved, establish that a chemiosmotic chip is not merely a sensor trick but a programmable computational substrate. Three signature experiments would serve as strong markers.

The first signature experiment is **an addressable gradient memory array with measured retention and refresh**. Concretely: an array of compartments (not just one) where you can write a spatial pattern of states (delta_pH and/or delta_psi), read it non-destructively, observe its decay, and restore it via a defined refresh protocol—repeatedly. The output should include retention distributions, refresh energy, and node health mapping. This would be the "DRAM moment" for chemiosmotic chips: proof that gradient state can be stored, addressed, and maintained as an engineered quantity.

The second signature experiment is **a reconfigurable canal network that implements two distinct transfer functions on demand**. Concretely: a protonic FPGA-like fabric where you can program the routing topology (by gates or light patterns), then show that the same physical substrate can behave as two different dynamical systems with measurably different input-

output behavior. For example, one configuration produces strong temporal integration and smoothing, another produces fast thresholded propagation. Demonstrate that you can switch between them reproducibly, with calibration, and that the system's behavior matches predicted network parameters. This would be the "FPGA moment": proof of programmability beyond a fixed device.

The third signature experiment is **a closed-loop learning demonstration in a gradient-native neuromorphic network**. Concretely: a small network where synaptic couplings are stored as local chemistry (buffering or permeability), where inputs drive activity, and where a local learning rule changes couplings in response to timing and a modulatory signal. Show that the system improves on a task over time—classification of temporal patterns, adaptive control of a simple plant, or stabilization of a drifting sensor—without external numerical weight updates. This would be the "learning moment": proof that chemiosmotic substrates can do more than hold gradients; they can adapt their coupling structure through physical learning.

If a research group can show these three—gradient memory with refresh, reconfigurable routing with distinct behaviors, and physical learning with measurable performance improvement—then the field has crossed a threshold. At that point, "chemiosmotic chip" becomes a legitimate computing platform concept rather than an evocative metaphor. The remaining work would be scaling, reliability, and integration—but the foundational claim would be established: gradients can be engineered into addressable, programmable, maintainable computational state.

11. Failure Modes, Limits, and Repairs

11.1 Leak-dominated scaling: why defects win unless architecture changes

The first hard limit that shows up as you scale chemiosmotic systems is brutally simple: leaks compound faster than intuition expects. A single well-sealed compartment can hold a gradient long enough to be impressive in a lab demo. A thousand compartments connected by a routing fabric is a different creature. The probability that *some* part of the system is leaky rises quickly with node count and interface count, and because gradients equalize, one bad leak can act like a short across an entire domain if isolation is not engineered.

Leak-dominated scaling is not only about membranes having pinholes. It is about *interfaces*. Every place a membrane touches a solid support, every electrode feedthrough, every gasket, every channel junction, every seal edge is an opportunity for an unintended shunt path. In CMOS, the number of failure sites also increases with scale, but the failure modes are mostly binary and can be screened. In chemiosmotic systems, leakage is analog, distributed, and time-varying, which makes it harder to treat as "pass/fail."

There is a deeper architectural issue: many naive designs assume that the chemical domain can behave like an electrical domain, with global rails and shared reservoirs. That assumption is lethal under leakage. Any shared rail becomes a conduit through which a defect bleeds away the energy of many nodes. As arrays scale, “global” becomes “fragile.” The architecture must therefore change to survive: power and state must become local, domains must be isolatable, and coupling must be explicitly gated.

This is why redundancy and mapping are not optional add-ons; they are scaling requirements. You must be able to characterize leakage per node (or per domain), assign roles accordingly, and quarantine defective elements. The fabric must support rerouting around weak nodes, and the control layer must schedule refresh with knowledge of heterogeneous retention times. Without these architectural responses, the system will be dominated by its worst leaks, and scale will only make it worse.

A key concept here is percolation. If leaks are random and sparse, you can isolate and survive. If leak pathways percolate through the fabric—through shared wet layers, poorly isolated interfaces, or widespread microdefects—then the system crosses a phase boundary where isolation is no longer possible. Successful scaling therefore requires not only better membranes but a layout that prevents percolation: compartmentalization, firebreak buffers, and hierarchical domains that limit the blast radius of a defect.

11.2 Drift and hysteresis: distinguishing useful memory from unbounded instability

Chemiosmotic computation lives on slow variables, and slow variables almost always bring drift and hysteresis. The naive response is to declare these fatal. The more interesting response is to separate *bounded hysteresis* (useful memory) from *unbounded drift* (instability that destroys interpretability). A mature system must draw that line explicitly and enforce it.

Bounded hysteresis is a form of analog memory: the system’s response depends on history, but within a controlled range. This can be exploited for neuromorphic weights, threshold tuning, and state retention between refresh cycles. It is useful when the hysteresis loop is stable, repeatable, and resettable. For example, a synaptic region that becomes more permeable after repeated coincident events and then slowly relaxes is not “drift”; it is a designed plasticity rule.

Unbounded drift is different. It is the gradual migration of baselines until the meaning of state variables is lost. Unbounded drift can be chemical (buffer depletion, net acidification), physical (membrane property changes, polymer swelling), or interfacial (surface charge aging, fouling). The signature of unbounded drift is that calibration no longer holds: the same write operation produces a different state change each day, and the system’s mapping between internal state and readout becomes nonstationary.

Distinguishing the two is partly measurement and partly design. Measurement: you must track baselines, retention curves, and response functions over time to see whether changes are reversible and bounded. Design: you must include explicit restore mechanisms that enforce bounds—re-zeroing to reservoirs, periodic neutrality correction, and guardrails that clamp pH and delta_psi excursions. A chemiosmotic system that cannot enforce bounds is not a computer; it is a chemical experiment.

One practical principle is to treat every state variable as having an allowed operating window and to include a watchdog that detects when it is approaching the boundary. This is the chemiosmotic analog of voltage and temperature monitors in SoCs. When a node drifts, the control layer intervenes: isolate, refresh, recondition, or retire. In this way, hysteresis becomes memory with a refresh strategy, not instability.

11.3 Biofouling and contamination: the hidden enemy of long-lived gradients

If leakage is the obvious enemy, contamination is the hidden one. Chemiosmotic chips are, by necessity, in contact with water and ions. Water is a universal solvent and a universal carrier of surprises: adsorbed organics, dissolved gases, trace metals, surfactants, microbial growth, and slow chemical reactions at surfaces. Over time, these alter permeability, surface charge, and sensor responses in ways that look like “mysterious drift” unless you treat contamination as a primary failure mode.

Biofouling is the extreme case: the growth of biofilms or microbial contamination that changes transport properties, blocks channels, and consumes or produces chemical species. Even in systems without biological components, biofouling can occur if the device is exposed to ambient environments and the wet layer is not sterile. For bio-derived membranes or proteins, the problem is amplified: proteins denature, membranes degrade, and biological materials provide footholds for microbes.

But contamination is not only biological. Ion contamination can change selectivity and conductivity dramatically. Trace multivalent ions can bind to membranes and polymers, altering their transport and gating behavior. Organic contamination can modify wettability and create unintended hydrophobic barriers or conductive films. Dissolved gases can form bubbles under thermal cycling or electrochemical actuation, introducing catastrophic channel blockage.

The painful truth is that contamination issues often appear late: the device works beautifully for hours or days and then degrades. This makes them especially dangerous for a field prone to short demonstrations. Long-lived gradients require long-lived chemical cleanliness. Therefore, any serious chemiosmotic chip program must include fluidic hygiene protocols as first-class engineering: controlled reservoirs, filtration, inert materials, preconditioning, and sealed packaging that prevents exchange with ambient contaminants.

In hybrid SoCs, the integrity problem also becomes a security problem. If external chemistry can perturb internal state, an attacker could spoof or bias the system. Even without malicious actors, environmental variation can act like a "spoof," making outputs unreliable. Therefore, contamination control and integrity monitoring belong together: treat unexpected chemical shifts as anomalies to be detected, not as noise to be averaged away.

11.4 Repair protocols: reconditioning membranes, recalibration, redundancy, and quarantine

The most important practical mindset shift is that chemiosmotic chips will not be "perfect devices." They will be systems that survive by repairing, recalibrating, and reconfiguring. Repairs occur at multiple levels: physical, chemical, and architectural.

Reconditioning membranes is the physical/chemical layer repair. If membranes foul or develop increased leakage, you may be able to partially restore performance by controlled cleaning cycles, rehydration protocols, or chemical washes that remove adsorbed contaminants without destroying functional components. In bio-derived systems, reconditioning might include gentle stabilization conditions that restore protein conformation or remove inhibitors. In polymer/oxide systems, reconditioning might include controlled pH cycling to reset surface charge states. The key is that reconditioning must be designed and tested as part of operation, not treated as an ad hoc rescue.

Recalibration is the measurement-layer repair. Sensors drift. Transducers age. The hybrid controller must periodically recalibrate against reference nodes and known perturbations. A practical protocol is built-in self-test: inject a known write pulse, measure response, update calibration coefficients. Recalibration should be multi-modal where possible: use both delta_pH sensing and delta_psi sensing, and cross-check against impedance signatures. This reduces the chance that a sensor drift will masquerade as a real state change.

Redundancy is the architectural repair strategy. You assume some nodes will degrade faster than others. You build extra nodes, extra channels, and extra reference domains. You then map the device, identify healthy regions, and allocate computation accordingly. Redundancy is not waste; it is how yield and lifetime become acceptable in a heterogeneous chemical substrate. Over time, as nodes fail, the system degrades gracefully rather than catastrophically.

Quarantine is the control strategy that makes redundancy effective. When a node becomes leaky, noisy, or contaminated, it must be isolatable. This requires switch fabrics that can disconnect a node from shared reservoirs and from coupling networks. Quarantine also requires that the system recognize failure signatures: abnormal retention decay, unexpected baseline drift, anomalous coupling responses. Once quarantined, a node can be retired, reconditioned offline, or used only for roles consistent with its degraded state (for example, fast transient dynamics where retention is less important).

Finally, the system needs a repair scheduler. In an operational device, reconditioning, recalibration, refresh, and quarantine actions must occur without destroying computation. This suggests an “always-on” maintenance layer that runs in the background: monitoring retention maps, adjusting refresh rates, recalibrating sensors, and periodically exercising nodes to detect early degradation. In other words, a chemiosmotic chip requires a maintenance operating system.

The payoff of embracing repair is that it turns a fragile substrate into an adaptive instrument. Instead of demanding that every node behave identically forever, you demand that the system remain interpretable and controllable over time. That is the realistic path to deployment: not perfect membranes, but managed membranes; not drift-free chemistry, but bounded drift with calibration; not defect-free arrays, but redundant fabrics with quarantine. This is the lesson electronics learned over decades, translated into the language of gradients: failure is inevitable, but collapse is optional if architecture is honest and maintenance is designed.

12. Conceptual Synthesis: Chemiosmosis as an Operating System

12.1 The OS metaphor: scheduling, refresh, isolation, and privilege levels

The chemiosmotic chip forces a conceptual shift that electronics has largely been able to avoid: the substrate is not stable enough to be “just hardware,” and the dynamics are too rich to be “just chemistry.” The right mental model is closer to an operating system than to a static circuit. Not because we want poetic metaphors, but because the decisive engineering problems—maintenance, arbitration, resource allocation, isolation, and fault management—look exactly like OS responsibilities, except the managed resources are gradients, membranes, and chemical domains.

Scheduling is the first OS-like function. Chemiosmotic systems are inherently time-dependent: writes take time, diffusion takes time, settling takes time, and reads can be disturbed by ongoing flux. Therefore, “running a program” means staging phases: write -> settle -> evaluate -> read -> refresh -> re-zero -> repeat. This is a scheduler’s job. It must allocate time windows to different domains, prevent conflicting operations from interfering, and exploit parallelism where possible (e.g., write many nodes optically while others settle). In CMOS, we can pretend time is merely clock ticks; in chemiosmosis, time is a physical process budget.

Refresh becomes the second OS function. As argued earlier, chemiosmotic state decays and drifts. Refresh is not optional background; it is part of correct operation. This is exactly the relationship DRAM has with refresh, but broadened: refresh includes neutrality correction, baseline re-zeroing, and recalibration of sensors. An OS for chemiosmosis must treat refresh as a scheduled maintenance interrupt: it happens periodically, it consumes resources, and it must

be coordinated so it does not corrupt computation. A chemiosmotic system that does not budget refresh is a system that will eventually "crash" chemically.

Isolation is the third OS function, and it is arguably the most important at scale. In a dense array, leaks and defects are inevitable. Isolation means the ability to disconnect a failing node, quarantine a contaminated domain, and prevent faults from propagating through shared reservoirs or coupling networks. This is OS-like in the same sense that memory protection and process isolation are OS functions: you do not allow one faulty process to corrupt all of RAM; you do not allow one leaky compartment to collapse a global gradient domain. The hardware must provide isolation primitives (valves, gates, firebreaks), and the operating layer must apply them based on diagnostics.

Privilege levels are the fourth OS function and provide a clean way to separate "compute" from "maintenance." In electronics, privileged operations are things like firmware updates, microcode, or management engines. In chemiosmotic systems, privileged operations are the ones that can change global baselines or reconfigure routing: opening reservoir connections, reconditioning membranes, executing deep re-zero cycles, or changing the mapping of nodes to roles. These operations must not be casually accessible to user-level computation because they can destroy state or destabilize the fabric. Therefore, the OS metaphor naturally suggests layered control:

- A **kernel** layer that manages physical integrity: hydration control, reservoir access, reconditioning, quarantine, and baseline stabilization.
- A **runtime** layer that schedules computation and refresh: evaluation windows, write/read phases, and calibration updates.
- A **user** layer that expresses tasks: input patterns, desired outputs, and high-level configurations (e.g., "run reservoir classifier," "run adaptive control loop").

This hierarchy matters because it makes explicit what chemiosmotic chips are really doing: they are maintaining a fragile, stateful physical medium in a narrow operating regime where computation is interpretable. The "OS" is not software in the usual sense; it is the integrated discipline of control logic, sensors, actuators, and maintenance policies.

12.2 Entrainment architectures: patterned drives and state locking in gradient space

Once we accept chemiosmosis as an OS-managed substrate, the next conceptual step is to recognize that the most natural way to "program" such a system is not through discrete instructions alone, but through **entrainment**: applying patterned drives that lock the system into stable trajectories in its state space.

Entrainment is a universal phenomenon in dynamical systems: a periodically forced oscillator can lock to the driving frequency; a coupled network can synchronize; a dissipative system can be shepherded into an attractor. Chemiosmotic networks—compartments connected by channels, with buffering and leakage—are dissipative dynamical systems par excellence. They have natural relaxation times, coupling strengths, and thresholds. When you drive them with patterned illumination, periodic gating, or modulated reservoir biases, you are not merely "writing bits." You are shaping the attractor landscape.

This suggests an architecture style where computation is realized as **state locking** rather than as Boolean evaluation. The OS applies a drive pattern (optical, electrical, chemical) that establishes a set of stable states or trajectories, and the system's input perturbs it. The readout is then not the instantaneous state of a single node but the system's convergence behavior: which attractor it falls into, how quickly it locks, what spatiotemporal pattern emerges.

Patterned drives can be used in several distinct ways:

- **Global entrainment:** a chip-wide periodic modulation that stabilizes baselines and imposes a common time reference. This is the chemiosmotic analog of a clock, but it is also a homeostatic stabilizer: the periodic drive can prevent drift by repeatedly re-centering states.
- **Regional entrainment:** different domains receive different drive patterns, effectively creating multiple "clock domains" or "mode domains" in gradient space. Some domains may be tuned for fast excitability, others for slow integration.
- **Task-specific entrainment:** drive patterns are chosen to make certain input classes separable. This resembles reservoir computing: the drive shapes the reservoir's response kernel.
- **Maintenance entrainment:** drive patterns that periodically flush, re-zero, or recondition without halting computation entirely—maintenance woven into dynamics rather than imposed as a stop-the-world refresh.

In such systems, dissipation is not the enemy. Dissipation is what makes entrainment possible: it allows the system to lose memory of irrelevant perturbations and converge to stable patterns. The OS's job is to tune dissipation: too little and the system becomes chaotic or overly persistent; too much and it becomes unresponsive. This is exactly the OS-like tension between responsiveness and stability, now expressed as physics.

Therefore, "programming" a chemiosmotic chip may often look like designing drive waveforms and coupling topologies rather than writing instruction sequences. The instruction set still exists

(write, gate, read, refresh), but it becomes subservient to a higher-level control strategy: shepherd the system into the right dynamical regime, then let the physics compute.

12.3 When analogy becomes physical: alignment-as-entrainment in a chemiosmotic substrate

At this point the OS metaphor stops being metaphorical and becomes physically literal. In chemiosmotic substrates, "alignment" is not only a conceptual goal (as in aligning representations or states). Alignment is a measurable dynamical act: the system's internal states $q(t)$ are driven toward target trajectories $\tau(t)$ by patterned forcing and coupling under dissipation. In other words, alignment is entrainment.

This is where the chemiosmotic chip offers a striking physical instantiation of an idea that, in purely digital systems, remains abstract. In a chemiosmotic medium, you can define a target gradient pattern or target trajectory—implemented as a drive pattern (illumination, gating schedule, reservoir bias)—and the system will either lock into it or fail to lock depending on its internal parameters, noise, and drift. The "truth" of the system's state is not a static bit pattern; it is whether the system stays in phase with the target dynamics under perturbation.

This gives a concrete meaning to "discipline" in the substrate. The OS enforces a discipline by repeatedly applying the drive that defines the operating manifold, and by using refresh and isolation to keep the system capable of locking. When the system drifts, it detunes; when it detunes, it loses lock; when it loses lock, outputs degrade. Repair then becomes retuning: re-zero baselines, restore buffer capacity, recondition membranes, recalibrate sensors, and bring the system back into a regime where entrainment is possible.

In this framework, there is also a clean separation between two types of control:

- **External entrainment** (the OS drive): a forcing pattern that defines what counts as "aligned operation" for the device.
- **Internal entrainment** (network coupling): the way compartments synchronize and stabilize each other through channels, thresholds, and buffering.

When these cooperate, the system becomes robust: local coupling resists small disturbances, and global drive prevents long-term drift. When they conflict—when coupling spreads faults or when global drive excites unstable modes—the system becomes fragile. Thus, the design problem becomes: choose coupling topologies and drive patterns that make desired modes attractors and undesired modes repellors, while keeping maintenance overhead tolerable.

This is also the point where a chemiosmotic chip begins to feel "more advanced" than the early protonics literature might suggest. The most interesting future systems will not be those that merely transport protons quickly. They will be those that treat gradients as a programmable dynamical field: a medium that can be entrained, locked, reset, and reconfigured. In such

systems, computation is the controlled emergence of stable trajectories under drive and dissipation. The OS metaphor captures this because it forces us to admit: the real work is not only the computation itself; it is the continuous governance of a physical medium so that computation remains meaningful.

So the synthesis is this: chemiosmosis is not just a power mechanism. It is a governance mechanism. It stores energy in gradients, routes it in space, and—when paired with scheduling, refresh, isolation, and entrainment—forms an operating system for physical computation. The chip runs not by pretending matter is static, but by continuously retuning matter so that its natural dynamics stay aligned with the intended attractors. That is the deep reason chemiosmotic computing is both difficult and promising: it asks engineering to do what biology has always done—compute by keeping a dissipative system in a narrow, living regime where structure persists.

13. Conclusion: Beyond Electronics Without Replacing Electronics

13.1 Restating the thesis: gradients as first-class computational state

The central claim of this paper can be stated cleanly: chemiosmotic gradients are not merely a way to power biology; they are a way to *represent and manipulate state*. If we treat gradients—pmf partitioned into delta_pH and delta_psi—as first-class computational variables, then "chemiosmotic chips" stop being a metaphor and become a coherent design space with primitives, architectures, metrics, and scaling pathways.

Electronics made computation cheap by reducing state to charge on capacitors and the controlled opening and closing of conduction channels. Chemiosmosis offers an alternate reduction: state as the distribution of chemical potential across boundaries, stored in buffered volumes and capacitive membranes, moved through gated channels, and restored by complementary push–pull mechanisms and explicit housekeeping. The resulting substrate is slower, leakier, and more context-dependent than CMOS—but it is also richer in the kinds of state it can hold and the ways it can couple to the physical world. It naturally supports memory-in-place, continuous-time dynamics, and computation that is not separable from sensing and actuation.

This is why "beyond electronics" does not mean "better electronics." It means a complementary substrate that extends what counts as computable state. In chemiosmotic media, "voltage" and "concentration" are not nuisances; they are the computational alphabet. The chip's instruction set becomes: write gradients, route gradients, sense gradients, couple gradients, and restore gradients. That is the thesis in operational form.

13.2 The practical takeaway: build hybrid stacks, prove signature demos, then scale

The practical path forward is also clear, and it is deliberately conservative. Start hybrid, because hybrids allow the chemiosmotic core to be treated as a managed physical medium rather than as a fragile novelty. Use electronics for timing, calibration, anomaly detection, and I/O. Use the chemiosmotic layer for what it does uniquely well: storing and shaping analog state in local gradient domains, and enabling dynamical computation through coupling and entrainment.

Then focus ruthlessly on decisive demonstrations, because this field is otherwise vulnerable to "one impressive plot" syndrome. The most credible progression is:

1. Demonstrate an addressable gradient memory array with measured retention, variability maps, and a working refresh protocol that can be repeated over many cycles. This establishes that gradients can be treated as engineered state, not transient chemistry.
2. Demonstrate reconfigurable routing that produces distinct, reproducible input–output behaviors from the same physical canal network. This establishes programmability at the topology level, not just at the device level.
3. Demonstrate closed-loop learning in a gradient-native neuromorphic network where weights are local chemistry and performance improves on a task over time under a local learning rule with homeostatic bounds. This establishes that the substrate can adapt, not merely store.

Only after these three signatures are real do you scale. And scaling should not be framed as "make it faster." It should be framed as "make it maintainable." Increase node count only as fast as isolation, metrology, redundancy, and repair protocols can keep the system interpretable. Treat defects as inevitable and map around them. Treat refresh as an OS function and schedule it explicitly. Treat packaging and cleanliness as computational integrity issues, not as afterthoughts.

If this sounds like a slower path than the rhetoric around "bio-inspired computing" often suggests, that is a feature. Chemiosmotic computation is a boundary science: it lives at interfaces—membrane/solid, wet/dry, chemical/electrical—and interfaces are where engineering succeeds only when it is disciplined.

13.3 Closing reflection: a boundary-driven physics of computation

There is a final reflection that unifies the technical story. Chemiosmosis teaches that computation is not only about what flows; it is about what *separates*. The membrane is not merely a container wall; it is the condition that makes gradients meaningful. The channel is not merely a conduit; it is a shaped constraint that determines how state propagates. The pump is

not merely an energy input; it is a writer that creates structured difference. The refresh cycle is not merely maintenance; it is the repeated reassertion of boundaries against entropy.

Electronics has its own boundary story—junctions, gate oxides, depletion regions—but it hides that story beneath abstraction. Chemiosmotic systems cannot hide it. Their computation is boundary-driven in the most explicit sense: boundaries create state, boundaries protect state, boundaries route state, and boundary failure is computation failure. This is why chemiosmotic chips force a new design attitude: one that treats integrity, isolation, and restoration as part of the computational model rather than as external reliability concerns.

If successful, chemiosmotic computing will not replace electronics. It will sit beside it as a new physical layer of state: slower, wetter, more analog, and more intimate with the world's chemistry—capable of tasks where living systems excel and where digital logic is awkward. The prize is not a faster calculator. The prize is a new kind of machine: one that computes by shaping gradients in space and time, governed by an operating discipline that keeps a dissipative medium locked within a narrow, interpretable regime. That is "beyond electronics" in its most literal sense: not abandoning charge, but adding gradient as another first-class way matter can remember, decide, and act.

14. References

14.1 Chemiosmosis and membrane bioenergetics foundations

- Mitchell, P. (1961). Coupling of phosphorylation to electron and hydrogen transfer by a chemi-osmotic type of mechanism. *Nature*, 191, 144–148. doi:10.1038/191144a0. Nature+1
- Nicholls, D. G., & Ferguson, S. J. (2013). *Bioenergetics 4*. Academic Press / Elsevier. Elsevier Shop+1
- Boyer, P. D. (1997). The ATP synthase — a splendid molecular machine. *Annual Review of Biochemistry*, 66, 717–749.
- Walker, J. E. (1998). ATP synthesis by rotary catalysis (Nobel lecture context). *Angewandte Chemie International Edition*, 37, 2308–2319.
- Junge, W., Lill, H., & Engelbrecht, S. (1997). ATP synthase: an electrochemical transducer with rotatory mechanics. *Trends in Biochemical Sciences*, 22, 420–423.
- Wikström, M. (1977). Proton pump coupled to cytochrome c oxidase in mitochondria. *Nature*, 266, 271–273.
- Harold, F. M. (1986). *The Vital Force: A Study of Bioenergetics*. W. H. Freeman.
- Alberts, B., et al. (latest ed.). Membrane transport and chemiosmotic coupling chapters. *Molecular Biology of the Cell*.

14.2 Proton transport physics and protonic materials

- Cukierman, S. (2006). Et tu, Grotthuss! and other unfinished stories. *Biochimica et Biophysica Acta (BBA) – Bioenergetics*, 1757, 876–885. sciencedirect.com
- Han, S. H., et al. (2022). Iontronics: aqueous ion-based engineering for bioinspired devices (review). *Current Protein & Peptide Science / AIP* (review PDF). AIP Publishing
- Chun, H., et al. (2015). Iontronics (review). *Advanced Materials* (or equivalent venue; PubMed record). PubMed
- Yao, X., et al. (2020). Protonic solid-state electrochemical synapse for physical neural networks. *Nature Communications*, 11, 3134. Nature
- Pati, S. P., et al. (2024). Real-time information processing via volatile resistance changes driven by proton dynamics. *Communications Materials*, 5, (article). Nature

- (Review) Proton conducting neuromorphic materials and devices. *Chemical Reviews* (2024). American Chemical Society Publications
- Kreuer, K.-D. (1997/2001). Proton conductivity: materials and mechanisms (classic reviews). *Annual Review of Materials Research* / *Solid State Ionics* (representative).
- Mauritz, K. A., & Moore, R. B. (2004). State of understanding of Nafion (structure/proton transport). *Chemical Reviews*, 104, 4535–4586.
- (Perspective) Proton transport in confined water / hydrated interfaces (representative reviews in *Chemical Reviews* / *Accounts of Chemical Research*).

14.3 Iontronics, electrochemical transistors, and neuromorphic devices

- Rivnay, J., Inal, S., Salleo, A., Owens, R. M., Berggren, M., & Malliaras, G. G. (2018). Organic electrochemical transistors. *Nature Reviews Materials*, 3, 17086. ADS+1
- Ohayon, D., Druet, V., & Inal, S. (2023). A guide for the characterization of organic electrochemical transistors and channel materials. *Chemical Society Reviews*, 52, (review). RSC Publishing
- Rashid, R. B., et al. (2021). Organic electrochemical transistors in bioelectronic circuits (review). *Current Opinion in Electrochemistry*, 30, 100773. sciencedirect.com
- Kim, M. K., et al. (2020). Emerging materials for neuromorphic devices and systems (includes electrochemical/three-terminal synapses). *Advanced Materials / related review*. PMC
- Zhang, F., et al. (2023). Recent progress in three-terminal artificial synapses (includes proton/ion coupled synaptic transistors). *npj Flexible Electronics*, 7, (review). Nature
- Lanza, M., et al. (2021). Standards for the characterization of endurance in resistive switching devices. *ACS Nano*, 15, 17268–17307. American Chemical Society Publications+1
- Schuman, C. D., et al. (2022). Opportunities for neuromorphic computing algorithms and applications (benchmarking context). *Nature Reviews Physics*, 4, (review). Nature
- Iyer, L. R., et al. (2021). Is Neuromorphic MNIST neuromorphic? (dataset/benchmark critique). *Frontiers in Neuroscience*, 15, 608567. Frontiers

14.4 Nanofluidics, membrane integration, and hybrid bioelectronic platforms

- Vlassiouk, I., et al. (2008). Nanofluidic ionic diodes: analytical vs numerical comparisons (rectification fundamentals). *ACS Nano*, 2, (paper). American Chemical Society Publications
- Liu, W., et al. (2024). Bioinspired carbon nanotube–based nanofluidic ionic transistors (voltage-gated ion channels/ionic transistor concepts). *PNAS / related open-access review*. PMC
- Sun, Y., et al. (2023). Electrokinetic transport phenomena in nanofluidics and their applications (review). *Electrophoresis*, 44, (review). analyticalsciencejournals.onlinelibrary.wiley.com
- Shi, D., et al. (2024). Nanofluidic systems for ion transport with tunable surface charges (review). *Frontiers in Lab on a Chip Technologies*, (review). Frontiers
- Atanasov, V., et al. (2005). A functional tethered lipid bilayer membrane on silicon oxide (electrical sealing + integration). *Biophysical Journal*, 89, 1780–1788. Cell
- Urisu, T., et al. (2005). Formation of high-resistance supported lipid bilayer on solid substrates for ion-channel biosensors. *Biosensors and Bioelectronics* (representative). sciencedirect.com
- Glazier, R., et al. (2017). Supported lipid bilayer platforms (review of SLB utility and integration). *Current Opinion / Methods-style review (open access)*. PMC
- Friedrich, T., et al. (2002). Proteorhodopsin is a light-driven proton pump with pH-dependent behavior. *Journal of Molecular Biology*, 321, 821–838. PubMed+1
- Walter, J. M., et al. (2007). Light-powering *E. coli* with proteorhodopsin (engineering light-driven pmf). *PNAS*, 104, 2408–2412. PNAS
- Stauffer, M., et al. (2020). Engineering and production of bacteriorhodopsin / purple membranes (review). *Biosensors*, 3(3), 51. MDPI

14.5 Benchmarking, reliability, and systems design methodologies

- Qureshi, M. K., et al. (2015). AVATAR: Variable-retention-time-aware refresh (DRAM retention as a benchmark analogy for refresh governance). *DSN* (paper). istc-cc.cmu.edu
- Cheng, W. K., et al. (2019). Retention-aware DRAM auto-refresh schemes (refresh interval norms and thermal dependence). *Electronics* / open-access review. PMC

- Lanza, M., et al. (2021). Standards for endurance characterization in resistive switching devices (methodology for honest endurance reporting). *ACS Nano*. American Chemical Society Publications+1
- Ohayon, D., et al. (2023). OECT characterization guide (repeatability, drift, and measurement hygiene). *Chemical Society Reviews*. RSC Publishing
- Song, Y., et al. (2024). Drift phenomena in gate-functionalized OECT biosensors (bias/solution/time drift modes). *Sensors / open-access* (paper). PMC
- Kulkarni, S., et al. (2021). Benchmarking neuromorphic and (reservoir-like) workloads (algorithm-based benchmarks; substrate-agnostic methodology). *OSTI report*. osti.gov
- "Reservoir computing benchmarks: a review, taxonomy, and critique" (2024). *arXiv* (benchmark task categorization). arXiv
- (Systems method) Fault isolation, redundancy, and yield management principles from VLSI test/reliability (standard texts in semiconductor reliability and fault-tolerant computing).

The author has published over 4,700 AI-assisted papers at: https://www.researchgate.net/profile/Douglas-Youvan/research . Google Scholar has indexed these preprints, and if you want a particular reference, the AI mode of a Google search (Gemini) has efficiently catalogued these preprints.